

VIDEO FRAME RENDERING METHOD AND APPARATUS, DEVICE, AND STORAGE MEDIUM

[0001] This application claims priority to Chinese Patent Application No. 202111010058.0 filed on August 31, 2021 and titled "VIDEO FRAME RENDERING METHOD, APPARATUS, AND DEVICE, AND STORAGE MEDIUM", which is incorporated by reference in its entirety.

FIELD

[0002] The present disclosure relates to the technical field of computers, and in particular to a video frame rendering method, apparatus, and device, and a storage medium.

BACKGROUND

[0003] With the development of cloud computing technology, users can complete, through cloud computing, tasks that are difficult to be completed by terminals. For example, in the field of cloud games, a user can play, through the cloud computing technology, a game that the terminal cannot run smoothly. Background processing related to the game is all completed by a cloud gaming server, and the terminal only needs to transmit control information to the cloud gaming server. The control information is used for controlling a game object in a game scene. The cloud gaming server can perform background processing based on the control information to obtain a video frame. The cloud gaming server transmits the video frame to the terminal. The terminal displays the video frame.

[0004] In the related technology, for a certain game object in the game scene, a display effect of the game object is configured in advance by a technician during a process of designing a game. For example, for a vehicle in the game scene, a color and a style of the vehicle are configured in advance by the technician.

SUMMARY

[0005] According to embodiments of the present disclosure, a video frame rendering method, apparatus, and device, and a storage medium are provided, so as to achieve a function of performing secondary rendering on a virtual object in a cloud application. Technical solutions of the present disclosure are as follows.

[0006] In an aspect, a video frame rendering method is provided. The method includes:

[0007] acquiring, by a server, a first video frame corresponding to a first terminal, the first video frame being a video frame obtained by rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene, and the controlled virtual object being a virtual object controlled by the first terminal;

[0008] rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to obtain a second video frame in a case that the first video frame displays the target virtual object of the first terminal, the target virtual object being a virtual object to be re-rendered by the first terminal; and

[0009] transmitting, by the server, the second video frame to the first terminal, the first terminal being configured to display the second video frame.

[0010] In an aspect, a video frame rendering system is provided. The system includes:

[0011] a first terminal, a first server, and a second server. The first terminal, the first server, and the second server are in communication connection with each other.

[0012] The first server is configured to render a target virtual scene from a perspective of a controlled virtual object in the target virtual scene to obtain a first video frame corresponding to the first terminal, and transmit the first video frame to the second server. The controlled virtual object is a virtual object controlled by the first terminal.

[0013] The second server is configured to receive the first video frame.

[0014] The second server is further configured to render the target virtual object in the first video frame based on a first parameter of the first terminal to obtain a second video frame in a case that the first video frame displays the target virtual object of the first terminal.

[0015] The second server is further configured to transmit the second video frame to the first terminal.

[0016] The first terminal is configured to display the second video frame in response to receiving the second video frame.

[0017] In an aspect, a video frame rendering apparatus is provided. The apparatus includes: a first video frame acquisition module, a rendering module and a transmitting module.

[0018] The first video frame acquisition module is configured to acquire a first video frame corresponding to a first terminal. The first video frame is a video frame obtained by rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene, and the controlled virtual object is a virtual object controlled by the first terminal.

[0019] The rendering module is configured to render a target virtual object in the first video frame based on a first parameter of the first terminal to obtain a second video frame in a case that the first video frame displays the target virtual object of the first terminal. The target virtual object is a virtual object that needs to be re-rendered by the first terminal.

[0020] The transmitting module is configured to transmit the second video frame to the first terminal. The first terminal is configured to display the second video frame.

[0021] In an aspect, a server is provided. The server includes one or more processors and one or more memories. The one or more memories store at least one computer program, and the computer program is loaded and executed by the one or more processors to implement the video frame rendering method.

[0022] In an aspect, a computer-readable storage medium is provided. The computer-readable storage medium stores at least one computer program, and the computer program is loaded and executed by a processor to implement the video frame rendering method.

[0023] In an aspect, a computer program product is provided. The computer program product includes a program code. The program code is executed by a processor to implement the video frame rendering method.

[0024] According to the technical solutions provided by the embodiments of the present disclosure, when it is desired to change a display effect of a certain virtual object, the first parameter is determined for the virtual object by the first terminal. Secondary rendering is performed on the first video frame based on the first parameter to obtain the second video frame. Compared with the first video frame, the display effect of the virtual object displayed in the second video frame is also the display effect configured by the first terminal for the virtual object. According to the above technical solutions, a function of performing secondary rendering on the virtual object is achieved in a cloud application, so that the user can quickly and efficiently adjust the display effect of the certain virtual object in the virtual scene, thereby extending a functional range of the cloud application and improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

BRIEF DESCRIPTION OF THE DRAWINGS

[0025] In order to describe the technical solutions in the embodiments of the present disclosure more clearly, the drawings required to be used in descriptions of the embodiments will be simply introduced below. Apparently, the drawings in the following descriptions show only some

embodiments of the present disclosure. Those skilled in the art may further obtain other drawings according to these drawings without creative work.

[0026] FIG. 1 is a schematic diagram of an implementation environment of a video frame rendering method according to an embodiment of the present disclosure.

[0027] FIG. 2 is a flowchart of a video frame rendering method according to an embodiment of the present disclosure.

[0028] FIG. 3 is a flowchart of a video frame rendering method according to an embodiment of the present disclosure.

[0029] FIG. 4 is a schematic diagram of an interface according to an embodiment of the present disclosure.

[0030] FIG. 5 is a schematic diagram of an interface according to an embodiment of the present disclosure.

[0031] FIG. 6 is a flowchart of a video frame rendering method according to an embodiment of the present disclosure.

[0032] FIG. 7 is a schematic diagram of an interface according to an embodiment of the present disclosure.

[0033] FIG. 8 is a schematic structural diagram of a video frame rendering apparatus according to an embodiment of the present disclosure.

[0034] FIG. 9 is a schematic structural diagram of a server according to an embodiment of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0035] To illustrate the objectives, technical solutions, and advantages of the present disclosure clearer, embodiments of the present disclosure are further described in detail with reference to the drawings.

[0036] The terms "first", "second", and the like in the present disclosure are used for distinguishing the same or similar items with substantially the same effects and functions. It is to be understood that the terms "first", "second", and "n-th" do not have a logical or chronological dependency on each other and do not limit the number and order of execution.

[0037] The term "at least one" in the present disclosure refers to one or more, and the meaning of "multiple" refers to two or more, for example, multiple reference face images refers to two or more reference face images.

[0038] A cloud application: it is a new application that changes a usage mode of a conventional application "local installation and local operation" into an "on-demand" service, and connects and manipulates a remote server cluster through Internet or a local area network to complete business logic or a computing task. The cloud application runs in the remote server cluster, but an interface of the cloud application is displayed on the terminal, which can reduce the operating cost of the terminal and greatly improve the work efficiency.

[0039] Cloud Gaming, also known as Gaming on Demand, is a cloud computing-based online gaming technology. Cloud gaming technology enables a Thin Client that has relatively limited graphics processing and data computing capabilities to run high quality games. In a cloud gaming scene, the game does not run in a player gaming terminal, but runs in a cloud server. The game scene is rendered into a video and audio stream by a cloud gaming server, and is transmitted to the player gaming terminal through a network. The player game terminal does not need to have powerful graphics computing and data processing capabilities, as long as the player game terminal has a basic streaming media playing capability and the capability of acquiring player input instructions and transmitting the same to the cloud gaming server.

[0040] A virtual scene: it is a virtual scene that is displayed (or provided) when an application is run. The virtual scene may be a real-world simulation environment, a semi-simulation semi-fictional virtual environment, or 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 the dimensions of the virtual scene are not limited in the embodiments of the present disclosure. For example, the virtual scene may include the sky, a land, a sea, and the like. The land may include an environmental element such as a desert and a city, and a user may control a virtual object to move in the virtual scene.

[0041] A virtual object: it refers to a movable object in the virtual scene. The movable object may be a virtual character, a virtual animal, an animation character, and the like, for example: a character, an animal, a plant, an oil bucket, a wall, and a stone displayed in the virtual scene. The virtual object may be a virtual avatar for representing a user in the virtual scene. Multiple virtual objects may be included in the virtual scene. Each virtual object has its own shape and volume in the virtual scene, and occupies a part of space in the virtual scene.

[0042] In some embodiments, the virtual object is a player character controlled by an operation on a client, or an artificial intelligence (AI) set in a virtual scene battle, or a non-player character (NPC) set in the virtual scene. In some embodiments, the virtual object is a virtual character competing in the virtual scene. In some embodiments, the quantity of virtual objects participating in an interaction in the virtual scene is preset or is dynamically determined based on the quantity of clients participating in the interaction.

[0043] Taking a shooting game as an example, a user can control a virtual object to freely fall, glide or open a parachute to fall, and the like in the sky of the virtual scene, to run, jump, crawl, bend forward, and the like on the land, and can also control the virtual object to swim, float or dive, and the like in the sea. Certainly, the user may also control the virtual object to ride on a virtual carrier to move in the virtual scene, for example, the virtual carrier may be a virtual vehicle, a virtual aircraft, or a virtual yacht. The above scene is merely taken as an example herein, and the scene is not limited in the embodiments of the present disclosure. The user may also control the interactions, such as fighting, of the virtual object with other virtual objects by means of an interaction prop. For example, the interaction prop may be a throwing interaction prop such as a virtual hand grenade, a virtual cluster mine, and a virtual sticky hand grenade (referred to as a "virtual sticky grenade"), and may also be a shooting interaction prop such as a virtual machine gun, a virtual hand gun, and a virtual rifle. The type of the interaction prop is not limited in the embodiments of the present disclosure.

[0044] An Android container: android is packaged into a container image and then is published through a standard container image. A carrier may be any container shell supporting open container initiative (OCI), so that the carrier may be easily maintained through kubernetes (k8s, a container orchestration engine). Thousands of server clusters can be easily deployed on a cloud by means of a powerful maintenance tool of k8s.

[0045] FIG. 1 is a schematic diagram of an implementation environment of a video frame rendering method according to an embodiment of the present disclosure. With reference to FIG. 1, the implementation environment may include a first terminal 110, a client server 120, a first server 130, a game server 140, and a second server 150. In some embodiments, the first terminal 110, the client server 120, the first server 130, the game server 140, and the second server 150 are nodes in a block chain system, and data transmitted between the first terminal 110, the client server 120, the first server 130, the game server 140, and the second server 150 is also stored on the block chain.

[0046] The first terminal 110 is connected to the client server 120 through a wireless network or a wired network. In an example, the first terminal 110 is a smart phone, a tablet computer, a

notebook computer, a desktop computer, a smart watch, or the like, but is not limited thereto. The first terminal 110 is installed with a client supporting virtual scene display, and runs the client.

[0047] The client server 120 is a server providing a service for a client. The client server 120 is an independent physical server, or a server cluster or a distributed system composed of multiple physical servers, or a server providing basic cloud computing services such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a content delivery network (CDN), big data, and an artificial intelligence platform. The first terminal 110 logs into the client server 120 through a running client. The client server 120 provides a service related to a user account, for example, providing a service related to user account verification, or providing a service for determining a cloud gaming duration corresponding to the user account, or providing a service for storing a personalized setting corresponding to the user account. The services provided by the client server 120 are not limited in the embodiments of the present disclosure. The client server 120 is an intermediary connecting the client running on the first terminal 110 and the first server 130.

[0048] The first server 130 is an independent physical server, or a server cluster or a distributed system composed of multiple physical servers, or a server providing basic cloud computing services such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a CDN, big data, and an artificial intelligence platform. The first server 130 is connected to a background service related to displaying a virtual scene, and the first terminal 110 is connected to the first server 130 through a wireless network or a wired network. The first terminal 110 can transmit control information to the first server 130. The control information is used for controlling a virtual object in the virtual scene. The first server 130 renders the virtual scene based on the control information. The first server 130 is connected to the client server 120 through a wireless network or a wired network. The first server 130 can acquire relevant information about a user account from the client server 120, and perform relevant operations such as game initialization based on the relevant information. In an embodiment, the first server 130 is a cloud server.

[0049] The game server 140 is an independent physical server, or a server cluster or a distributed system composed of multiple physical servers, or a server providing basic cloud computing services such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a CDN, big data, and an artificial intelligence platform. The game server 140 is connected to the first server 120 through a wireless network or a wired network. The game

server 140 stores information related to a game character, such as friends of the game character, an address book, a level of the game character, and a name of the game character. The first server 130 can acquire the information related to the game character from the game server 140. One user account may correspond to multiple game characters. The client server 120 is configured to store the information related to the user account. The game server 140 is configured to store the information related to the game character.

[0050] The second server 150 is an independent physical server, or a server cluster or a distributed system composed of multiple physical servers, or a server providing basic cloud computing services such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a CDN, big data, and an artificial intelligence platform. The second server 150 provides a service related to secondary rendering of the virtual scene. The second server 150 is connected to the first server 120 through a wireless network or a wired network. The first server 120 can transmit a video frame after the virtual scene is rendered to the second server 150. The second server 150 performs secondary rendering on the video frame, so as to realize personalized processing of a user. The second server 150 can transmit the video frame subjected to the secondary rendering to the first terminal 110, and the first terminal 110 displays the video frame subjected to the secondary rendering. In addition, the second server 150 is connected to the client server 130 through a wireless network or a wired network. The second server 150 can acquire a personalized setting corresponding to a user account from the client server 130, and perform secondary rendering on the video frame transmitted by the first server 120 based on the personalized setting to obtain the video frame subjected to the secondary rendering. In some embodiments, the second server 150 is integrated into the first server 130. The function of the second server 150 is implemented by a system running on the first server 130. In an embodiment, the second server 150 may be a scene management server.

[0051] A person skilled in the art may know that the quantity of the above first terminals may be more or less. For example, there is only one first terminal, there are dozens or hundreds of first terminals, or there are more first terminals. In a case that the quantity of the first terminal is more than one, the above implementation environment further includes other first terminals. The quantity and a type of the first terminal are not limited in the embodiments of the present disclosure.

[0052] After an application scene of the video frame rendering method provided by the embodiments of the present disclosure is described, the application scene of the video frame rendering method provided by the embodiments of the present disclosure will be described below in combination with the above implementation environment. In the following description process, the

first terminal is the first terminal 110 in the above implementation environment, the client server is the client server 120 in the above implementation environment, the first server is the first server 130 in the above implementation environment, and the game server is the game server 140 in the above implementation environment. The second server is the second server 150 in the above implementation environment.

[0053] The video frame rendering method provided by the embodiments of the present disclosure may be applied in various types of cloud gaming scenes, such as in a first-person shooting (FPS) game, in a third-person shooting (TPS) game, in a multiplayer online battle arena (MOBA), or in a chess game or an auto chess game. The application scenes are not limited in the embodiments of this application.

[0054] It is assumed that the video frame rendering method provided by the embodiments of the present disclosure is applied in the FPS game, for example. A user starts a cloud gaming client on a first terminal, and logs in a user account in the cloud gaming client. That is to say, the user inputs the user account and a corresponding password in the cloud gaming client, and clicks a login control to perform login. In response to detecting a click operation on the login control, the first terminal transmits a login request to the client server. The login request carries the user account and the corresponding password. After receiving the login request, the client server acquires the user account and the corresponding password from the login request, and verifies the user account and the corresponding password. After verifying the user account and the corresponding password, the client server transmits login success information to the first terminal. After receiving the login success information, the first terminal transmits a cloud gaming acquisition request to the client server. The cloud gaming acquisition request carries the user account. After receiving the cloud gaming acquisition request, the client server performs query based on the user account carried in the cloud gaming acquisition request, acquires multiple cloud games corresponding to the user account, and transmits identifiers of the multiple cloud games to the first terminal. The first terminal presents the identifiers of the multiple cloud games in the cloud gaming client. The user selects, through the first terminal, an identifier of the FPS game to be played from the identifiers of the multiple cloud games displayed in the cloud gaming client, that is, selects the FPS game to be played. After the user selects the FPS game in the cloud gaming client, the first terminal transmits a game start instruction to the client server. The game start instruction carries the user account, the identifier of the FPS game, and hardware information of the first terminal. The hardware information of the first terminal includes a screen resolution of the first terminal, a model of the first terminal, and the like. Content of the game instruction is not limited in the embodiments of the present disclosure. After receiving the game start instruction, the client server transmits the game start instruction to the first server. After receiving the

game start instruction, the first server acquires the user account, the identifier of the FPS game, and the hardware information of the first terminal from the game start instruction. The first server initializes the FPS game based on the hardware information of the first terminal to realize matching between a rendered game picture and the first terminal, and transmits the user account to the game server corresponding to the FPS game. After receiving the user account, the game server transmits information corresponding to the game account to the first server. The first server starts the FPS game based on the information corresponding to the game account. In a process of running the FPS game, the user can control a controlled virtual object in the FPS to move through the first terminal. That is, the first terminal transmits control information of the controlled virtual object to the first server. The first server renders the virtual scene of the FPS based on the control information to obtain a first video frame. The first server transmits the first video frame to the second server. If the user personalizes a certain virtual object in the FPS game before the game starts, for example, sets a color of a vehicle in the FPS game to be red, then the second server can perform image recognition on the first video frame to determine whether the first video frame includes the vehicle. If the second server determines that the first video frame displays the vehicle, a first parameter of the vehicle is acquired from the client server. The first parameter is determined by the client server based on a parameter set for the vehicle by the user. The second server renders the vehicle in the first video frame based on the first parameter to obtain a second video frame, and transmits the second video frame to the first terminal. The user can view the second video frame through the first terminal. If a technician configures the color of the vehicle to be blue in the FPS game, the color of the vehicle can be adjusted to red set by the user through the above steps, thereby achieving personalized configuration for the vehicle.

[0055] The above is described by taking the processing of the first video frame as an example. In the process of running the FPS game, there are multiple continuous video frames. The multiple video frames are all processed in the above mode for displaying by the first terminal.

[0056] In addition, for a MOBA game, a TPS game, a chess game, and an auto chess game, the above steps can all be used for processing. For example, in the MOBA game, the user can set a tree in the MOBA game to be a peach blossom through the first terminal, and then the second server can recognize the tree from a first video frame, perform secondary rendering on the tree to obtain the peach blossom and thus to obtain a second video frame, and transmit the second video frame to the first terminal for displaying.

[0057] Furthermore, the video frame rendering method provided by the embodiments of the present disclosure can also be applied to other types of cloud games besides the above FPS game,

MOBA game, chess game, or auto chess game. Types of the cloud games are not limited in the embodiments of the present disclosure.

[0058] After introducing an implementation environment and an application scene of the video frame rendering method provided by the embodiments of the present disclosure, the video frame rendering method provided by the embodiments of the present disclosure will be described below.

[0059] FIG. 2 is a flowchart of a video frame rendering method according to an embodiment of the present disclosure. With reference to figure 2, it is assumed that the method is performed by a second server. The method includes the following steps 201 to 203.

[0060] In step 201, a second server acquires a first video frame corresponding to a first terminal. The first video frame is a video frame obtained by rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene, and the controlled virtual object is a virtual object controlled by the first terminal.

[0061] The operation of rendering the target virtual scene from the perspective of the controlled virtual object in the target virtual scene includes: rendering a picture observed by the controlled virtual object in the target virtual scene. Since the controlled virtual object is a virtual object controlled by the first terminal, the picture observed by the controlled virtual object in the target virtual scene can simulate a picture seen by a user of the first terminal in the target virtual scene.

[0062] Different first terminals have different controlled virtual objects. In the same target virtual scene, pictures observed by different controlled virtual objects should be different, and therefore pictures displayed by different first terminals should also be different. For this reason, the target virtual scene needs to be rendered from the perspective of the controlled virtual object of the first terminal, and the obtained video frame corresponds to the first terminal.

[0063] In an example, the first video frame is rendered by the first server and then is transmitted to the second server.

[0064] In step 202, the second server renders the target virtual object in the first video frame based on a first parameter of the first terminal to obtain a second video frame, in a case that the first video frame displays the target virtual object of the first terminal.

[0065] The target virtual object is a virtual object set by the first terminal, and is a virtual object that needs to be re-rendered by the first terminal. For example, the target virtual object is a virtual object selected by the user through the first terminal, a virtual object selected by the first terminal according to a position of the controlled virtual object, or a virtual object set in other modes.

The first parameter is a first parameter of the first terminal, for example, the first parameter is a first parameter determined by the user through the first terminal, or a first parameter set by default by the first terminal. The first parameter is used for re-rendering the target virtual object in the first video frame to obtain the second video frame. Since both the target virtual object and the first parameter are configured through the first terminal, and are not affected by other terminals, thereby achieving personalized configuration for a virtual scene. When it is desired to change a display effect of a certain virtual object in the virtual scene, configuration can be performed directly through the first terminal without changing underlying code and files by a technician, thereby improving the processing efficiency.

[0066] In step 203, the second server transmits the second video frame to the first terminal, and the first terminal is configured to display the second video frame.

[0067] In an example, the video frame rendering method provided by the embodiments of the present disclosure is applied in a cloud gaming scene, and processes of rendering the virtual scene and rendering a video frame are both implemented in a cloud. The cloud here is a generic term of the second server (such as a scene management server), the first server (such as a cloud server), and other relevant servers. After performing secondary rendering on the first video frame, the second server transmits the obtained second video frame to the first terminal. The first terminal directly displays the second video frame without performing background processing, thereby improving the processing efficiency.

[0068] In the above embodiment of the present disclosure, the method is performed by the second server as an example. In other embodiments, the method may be performed by other servers.

[0069] According to the technical solutions provided by the embodiments of the present disclosure, when it is desired to change a display effect of a certain virtual object, the first parameter is determined for the virtual object by the first terminal. Secondary rendering is performed on the first video frame based on the first parameter to obtain the second video frame. Compared with the first video frame, the display effect of the virtual object displayed in the second video frame is also the display effect configured by the first terminal for the virtual object. According to the above technical solutions, a function of performing secondary rendering on the virtual object is achieved in a cloud application, so that the user can quickly and efficiently adjust the display effect of the certain virtual object in the virtual scene, thereby extending a functional range of the cloud application and improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

[0070] Through the above steps 201 to 203, the embodiments of the present disclosure are briefly described. The technical solutions provided by the embodiments of the present disclosure will be described more clearly below in combination with some examples. Taking the method performed by the second server as an example, with reference to FIG. 3, the method includes the following steps 301 to 308.

[0071] In step 301, a second server acquires a first video frame corresponding to a first terminal. The first video frame is a video frame obtained by rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene, and the controlled virtual object is a virtual object controlled by the first terminal.

[0072] In an embodiment, in a game scene, the target virtual scene is a game scene of cloud gaming, and the perspective of the controlled virtual object is a perspective of a virtual camera of the controlled virtual object. In an FPS game, the virtual camera of the controlled virtual object is located at the head of the controlled virtual object. When a user controls the controlled virtual object to move in the target virtual scene through the first terminal, the virtual camera also moves along with the movement of the controlled virtual object, and a picture captured by the virtual camera is a picture observed by the controlled virtual object in the target virtual scene. In a TPS game, the virtual camera of the controlled virtual object is located above the controlled virtual object. When the user controls the controlled virtual object to move in the target virtual scene through the first terminal, the virtual camera also moves along with the movement of the controlled virtual object, and a picture captured by the virtual camera is a picture observed from a position above the controlled virtual object. In a cloud gaming scene, a picture captured by the virtual camera is rendered by a cloud gaming server. Since there are multiple pictures continuous in timing during gaming, the multiple pictures are also referred to as a video frame. Similarly, in other scenes, the first server may capture a target virtual scene through the virtual camera to obtain a picture, and the captured picture is also referred to as a video frame. In an embodiment, the first server is a cloud server.

[0073] In one possible implementation, the first terminal transmits control information for the controlled virtual object to the first server. After receiving the control information, the first server determines a perspective of the controlled virtual object in the target virtual scene based on the control information. The first server renders the target virtual scene from the perspective of the controlled virtual object to obtain a first video frame. The first video frame is a video frame corresponding to the first terminal. The first server transmits the first video frame to the second server. The second server acquires the first video frame. The control information of the controlled virtual object is used for changing a position, an orientation, and an action of the controlled virtual object in the target

scene. For example, the control information can control the controlled virtual object to move back, forth, left, and right in the target virtual scene, can control the controlled virtual object to rotate left or right in the target virtual scene, or can control the controlled virtual object to perform actions such as squatting, creeping, and using a virtual prop in the target virtual scene. Practically, when the first server controls the controlled virtual object to move or perform an action in the target virtual scene based on the control information, the virtual camera bound to the controlled virtual object also moves along with the movement of the controlled virtual object. The movement or the action performed by the controlled virtual object results in a change in the perspective of the controlled virtual object for observing the target virtual scene, and the virtual camera bound to the controlled virtual object can record this change. In an embodiment, the first server is a scene management server.

[0074] In this implementation, the control information of the controlled virtual object is transmitted to the first server through the first terminal. The first server can determine a perspective of the controlled virtual object in the target virtual scene based on the control information, and render the target virtual scene from the perspective of the controlled virtual object in the target virtual scene to obtain a first video frame, without rendering by the first terminal, thereby improving the rendering efficiency. The second server directly performs subsequent processing on the first video frame.

[0075] For example, the first terminal displays the target virtual scene. The controlled virtual object is displayed in the target virtual scene. A long connection is established between the first terminal and the first server. In response to an operation on the controlled virtual object, the first terminal transmits control information corresponding to the operation to the first server. The long connection means that multiple data packets can be transmitted continuously on one connection. During a connection maintenance period, if there is no data packet is transmitted, the first terminal and the first server need to transmit a link detection packet. In some embodiments, the operation on the controlled virtual object includes a click operation and a drag operation. The click operation is a click operation on an operation control, such as a click operation on a fire control in the FPS game. The drag operation is a drag operation on the operation control, on the controlled virtual object, or the like, such as a drag operation on a skill control in the MOBA game. After receiving the control information, the first server determines a position, a posture, and an action of the controlled virtual object in the target virtual scene based on the control information. The first server determines the perspective of the controlled virtual object based on the position, the posture, and the action of the controlled virtual object in the target virtual scene. The first server renders the target virtual scene from the perspective of the controlled virtual object to obtain a first video frame corresponding to the first terminal. The first video frame is a video frame rendered based on the control information transmitted by the first terminal. For example, if the operation on the controlled virtual object by the

user is a click-to-squat control, the control information transmitted by the first terminal to the first server is used for controlling the controlled virtual object to squat. The first server controls the controlled virtual object to squat in the target virtual scene, and determines a perspective of the controlled virtual object after squatting. The target virtual scene is rendered from the perspective of the controlled virtual object after squatting to obtain the first video frame. The first server transmits the first video frame to the second server. The second server acquires the first video frame.

[0076] In step 302, the second server performs image recognition on the first video frame, to determine a type of the first video frame.

[0077] In one possible implementation, the second server performs detection in the first video frame by using a template image of the target virtual object. In response to detecting that there is a region that matches the template image of the target virtual object in the first video frame, the second server determines a type of the first video frame as a first type. The first type indicates that the first video frame displays the target virtual object. In response to detecting that there is no region that matches the template image of the target virtual object in the first video frame, the second server determines the type of the first video frame as a second type. The second type indicates that the first video frame does not display the target virtual object. An identifier of the target virtual object is transmitted to the second server by the client server at the beginning. The second server can determine, based on the identifier of the target virtual object, the target virtual object that needs to be recognized. For example, the second server searches for the template image stored correspondingly to the identifier based on the identifier of the target virtual object, that is, the template image of the target virtual object. Subsequently, detection is performed by using the template image when the video frame corresponding to the first terminal is acquired each time.

[0078] The target virtual object is a virtual object set by the first terminal. The target virtual object is any virtual object in the target virtual scene. For example, the target virtual object is a virtual vehicle in the target virtual scene, the target virtual object is a virtual tree in the target virtual scene, the target virtual object is a virtual house in a virtual scene, or the target virtual object is a virtual gun. The type of the target virtual object is not limited in the embodiments of the present disclosure.

[0079] In this implementation, the second server can determine the type of the first video frame by template matching, that is, determine whether a target virtual object is displayed in the first video frame. Since a template matching speed is relatively quick, the efficiency is relatively high when the type of the first video frame is determined by template matching.

[0080] The above implementation is described below by two examples.

[0081] Example 1, the second server acquires the similarities between the template image of the target virtual object and multiple regions in the first video frame by using the template image of the target virtual object. In a case that there is a region of which the similarity with the template image of the target virtual object satisfies a target similarity condition among the multiple regions, the second server determines that the region matches the template image of the target virtual object, and determines the type of the first video frame as the first type. In a case that none of the similarities between the multiple regions and the template image of the target virtual object satisfies the target similarity condition, the second server determines that there is no region that matches the template image of the target virtual object in the first video frame, and determines the type of the first video frame as the second type.

[0082] For example, the second server acquires the similarities between the template image of the target virtual object and multiple regions on the first video frame by sliding the template image of the target virtual object on the first video frame. The multiple regions are regions covered when the template image of the target virtual object slides on the first video frame. In some embodiments, the second server can determine the similarities between the template image of the target virtual object and the multiple regions by using a color value similarity or a gray value similarity. The manner for determining the similarity is not limited in the embodiments of the present disclosure. In a case that there is a region of which a similarity with the template image of the target virtual object is greater than or equal to a similarity threshold among the multiple regions, the second server determines that the region matches the template image of the target virtual object, and determines the type of the first video frame as the first type, that is, the first video frame displays the target virtual object. In a case that each of the similarities between the multiple regions and the template image of the target virtual object is less than the similarity threshold, the second server determines that there is no region that matches the template image of the target virtual object in the first video frame, and determines the type of the first video frame as the second type, that is, the first video frame does not display the target virtual object.

[0083] Example 2, the second server performs scaling processing on the template image of the target virtual object to obtain template images with multiple sizes of the target virtual object. The second server acquires the similarities between the template image of the target virtual object and multiple regions on the first video frame based on the template images with multiple sizes. The multiple regions include regions with different sizes. In a case that there is a region of which the similarity with the template image of the target virtual object satisfies a target similarity condition among the multiple regions, the second server determines that the region matches the template image of the target virtual object, and determines the type of the first video frame as the first type. In a case

that none of the similarities between the multiple regions and the template image of the target virtual object satisfies the target similarity condition, the second server determines that there is no region that matches the template image of the target virtual object in the first video frame, and determines the type of the first video frame as the second type.

[0084] Since the controlled virtual object moves in the target virtual scene, a distance between the controlled virtual object and the target virtual object may be different in different video frames, and thus the size of the target virtual object may also be different in different video frames. The template image of the target virtual object is scaled into the template images with multiple sizes to perform template matching, thereby improving the accuracy of the template matching.

[0085] For example, the second server performs scaling processing on the template image of the target virtual object to obtain the template images with multiple sizes of the target virtual object. The second server controls the template images with the multiple sizes to slide on the first video frame to acquire the similarities between the template images with the multiple sizes and the multiple regions on the first video frame. The multiple regions are regions covered when the template image of the target virtual object slides on the first video frame. In some embodiments, the second server can determine the similarities between the template images with the multiple sizes and the multiple regions by using a color value similarity or a gray value similarity. The manner for determining the similarity is not limited in the embodiments of the present disclosure. In a case that there is a region of which a similarity with any of the template images with the multiple sizes is greater than or equal to the similarity threshold among the multiple regions, the second server determines that the region matches the template image of the target virtual object, and determines the type of the first video frame as the first type, that is, the first video frame displays the target virtual object. In a case that each of the similarities between the multiple regions and the template image of the target virtual object is less than the similarity threshold, the second server determines that there is no region that matches the template image of the target virtual object in the first video frame, and determines the type of the first video frame as the second type, that is, the first video frame does not display the target virtual object.

[0086] In one possible implementation, the second server inputs the first video frame into an image recognition model. The second server performs feature extraction and classification on the first video frame through the image recognition model, to output the type of the first video frame.

[0087] The image recognition model is obtained by training based on a sample video frame set. The sample video frame set includes a positive sample video frame and a negative sample video frame. The positive sample video frame is a video frame displaying a target virtual object, and the

negative sample video frame is a video frame not displaying a target virtual object. The image recognition model trained by the sample video frame set can determine whether a target virtual object is displayed in the video frame. In some embodiments, the image recognition model is trained in advance by the second server. The type of the video frame can be directly determined by using the image recognition model during gaming.

[0088] The above implementation is described below by two examples.

[0089] Example 1, the second server inputs the first video frame into the image recognition model. The second server performs feature extraction and full connection processing on the first video frame through the image recognition model to obtain a probability corresponding to the first video frame. The probability is a probability that the first video frame displays the target virtual object. In response to the probability being greater than or equal to a probability threshold, the second server determines the type of the first video frame as a first type. The first type indicates that the first video frame displays the target virtual object. In response to the probability being less than the probability threshold value, the second server determines the type of the first video frame as a second type. The second type indicates that the first video frame does not display the target virtual object.

[0090] For example, the second server inputs the first video frame into the image recognition model, and performs convolution processing on the first video frame through the image recognition model to obtain a feature map of the first video frame. The second server performs full connection processing and normalization processing on the feature map of the first video frame through the image recognition model to obtain a probability distribution corresponding to the first video frame. The normalization processing may be performed by using an S-shaped growth curve (Sigmoid) or a Softmax function. The adopted functions are not limited in the embodiments of the present disclosure. In response to the probability corresponding to the first type in the probability distribution being greater than or equal to the probability threshold, the second server determines the type of the first video frame as the first type, that is, the first video frame displays the target virtual object. In response to the probability corresponding to the second type in the probability distribution being greater than or equal to the probability threshold, the second server determines the type of the first video frame as the second type, that is, the first video frame does not display the target virtual object.

[0091] Example 2, the second server divides the first video frame into multiple image blocks, and inputs the multiple image blocks into the image recognition model. The second server performs feature extraction and full connection processing on the multiple image blocks through the image recognition model to obtain multiple probabilities respectively corresponding to the multiple image blocks. The probability is a probability that the image block includes the target virtual object. In

response to any of the multiple probabilities being greater than or equal to a probability threshold, the second server determines the type of the first video frame as the first type. The first type indicates that the first video frame displays the target virtual object. In response to each of the multiple probabilities being less than the probability threshold, the second server determines the type of the first video frame as the second type. The second type indicates that the first video frame does not display the target virtual object.

[0092] For example, the second server divides the first video frame into multiple image blocks with the same size, inputs the multiple image blocks into the image recognition model, and performs convolution processing on the multiple image blocks through the image recognition model to obtain feature maps of the multiple image blocks. The second server performs full connection processing and normalization processing on the feature maps of the multiple image blocks through the image recognition model to obtain multiple probabilities corresponding to the multiple image blocks. The probability is a probability that the corresponding image block displays the target virtual object. The normalization processing may be performed by using an S-shaped growth curve (Sigmoid) or a Softmax function. The adopted function is not limited in the embodiments of the present disclosure. In response to any of the probabilities being greater than or equal to the probability threshold, the second server determines the type of the first video frame as the first type, that is, the first video frame displays the target virtual object. In response to each of the multiple probabilities being less than the probability threshold, the second server determines the type of the first video frame as the second type, that is, the first video frame does not display the target virtual object.

[0093] After step 302, the second server performs the following step 303 in a case that the type indicates that the first video frame displays the target virtual object; and the second server performs the following step 304 in a case that the type indicates that the first video frame does not display the target virtual object.

[0094] In step 303, the second server renders the target virtual object in the first video frame based on the first parameter of the first terminal to obtain a second video frame, in a case that the type indicates that the first video frame displays the target virtual object of the first terminal.

[0095] In one possible implementation, the second server acquires a first rendering parameter in a case that the first video frame displays the target virtual object of the first terminal. The first rendering parameter is a rendering parameter corresponding to a first angle and a first distance in the first parameter. The first angle indicates an angle between the controlled virtual object and the target virtual object, and the first distance indicates a distance between the controlled virtual object and the

target virtual object. The second server renders the target virtual object in the first video frame based on the first rendering parameter to obtain the second video frame.

[0096] In this implementation, the second server can perform secondary rendering on the target virtual object based on the first rendering parameter to change the display effect of the target virtual object, in a case that the first video frame displays the target virtual object. During gaming, the distance and angle between the controlled virtual object and the target virtual object may change at any time. Different distances and angles correspond to different rendering parameters. The second server can directly acquire the first rendering parameter from the first parameter based on the first angle and the first distance, and is unnecessary to acquire the first rendering parameter by performing a secondary operation based on the first angle and the first distance, thereby improving the processing efficiency.

[0097] For example, the second server acquires the first angle and the first distance between the controlled virtual object and the target virtual object from the first server, in a case that the first video frame displays the target virtual object of the first terminal. The second server transmits the first angle, the first distance, and an identifier of the target virtual object to the client server. After receiving the first angle, the first distance, and the identifier of the target virtual object, the client server determines a first parameter based on the identifier of the target virtual object. The client server performs searches the first parameter based on the first angle and the first distance to obtain the first rendering parameter, and transmits the first rendering parameter to the second server. The second server determines target pixel values of multiple target pixel points of the target virtual object in the first video frame based on the first rendering parameter. The second server updates the pixel values of the multiple target pixel points in the first video frame by using the target pixel values to obtain the second video frame.

[0098] The target pixel points refer to pixel points of the target virtual object in the first video frame. The target pixel points have pixel values in the first video frame. The target pixel values refer to the pixel values that need to be obtained by updating the target pixel points. Therefore, original pixel values of the target pixel points are updated by using the target pixel values.

[0099] The angle between the controlled virtual object and the target virtual object may be an angle between an orientation of the controlled virtual object and an orientation of the target virtual object. For example, if the controlled virtual object is a game character holding a virtual gun in a game scene, the orientation of the controlled virtual object is an aiming direction of the virtual gun. If the target virtual object is a virtual vehicle, the orientation of the target virtual object is right ahead the virtual vehicle. The distance between the controlled virtual object and the target virtual object is

a distance between a center point of the controlled virtual object and a center point of the target virtual object.

[0100] In one possible implementation, the second server acquires the first rendering parameter and the second rendering parameter from the first parameter, in a case that the first video frame displays the target virtual object of the first terminal. The second server renders the target virtual object and the controlled virtual object in the first video frame respectively by using the first rendering parameter and the second rendering parameter to obtain the second video frame.

[0101] In this implementation, since the first parameter is a parameter of the first terminal, the first parameter carries both the first rendering parameter for rendering the target virtual object and the second rendering parameter for rendering the controlled virtual object. This indicates that the user can not only perform secondary rendering on the target virtual object in the virtual scene, but also perform secondary rendering on the controlled virtual object controlled by the first terminal, thereby improving the personalization and playability.

[0102] For example, the second server acquires the first angle and the first distance between the controlled virtual object and the target virtual object from the first server, in a case that the first video frame displays the target virtual object of the first terminal. The second server transmits the first angle, the first distance, and an identifier of the target virtual object to the client server. After receiving the first angle, the first distance, and the identifier of the target virtual object, the client server determines a first parameter based on the identifier of the target virtual object. The client server searches the first parameter based on the first angle and the first distance to obtain the first rendering parameter and the second rendering parameter, and transmits the first rendering parameter and the second rendering parameter to the second server. After receiving the first rendering parameter and the second rendering parameter, the second server renders the target virtual object by using the first rendering parameter, and renders the controlled virtual object by using the second rendering parameter in the first video frame, to obtain a second video frame.

[0103] For example, a display effect of the target virtual object can be personalized and a display effect of the controlled virtual object controlled by the user can be personalized when displaying the target virtual object, by the above implementation. Taking the target virtual object being a virtual vehicle as an example, if the user wants to set a color of the virtual vehicle to be red, and also wants to display a color of clothing of the game character (the virtual object) controlled by the first terminal as blue when the virtual vehicle appears in the target virtual scene, the first rendering parameter and the second rendering parameter can be configured in the first parameter by the above implementation. When the first video frame displays the target virtual object, the second server can

perform secondary rendering on the controlled virtual object by using the first rendering parameter while performing secondary rendering on the target virtual object by using the first rendering parameter, thereby improving the personalization.

[0104] In one possible implementation, the second server determines a position of the controlled virtual object in the target virtual scene in a case that the first video frame displays the target virtual object of the first terminal. The second server renders the target virtual object in the first video frame based on the first parameter to obtain the second video frame, in a case that the controlled virtual object is located in a target sub-scene of the target virtual scene. The second server determines the first video frame as the second video frame in a case that the controlled virtual object is not located in the target sub-scene of the target virtual scene.

[0105] The target virtual scene includes multiple sub-scenes, and the multiple sub-scenes constitute the target virtual scene. For example, the multiple sub-scenes include a virtual social scene and a virtual battle scene. The virtual social scene includes a virtual fishing scene, a virtual chat room, a virtual dance room, or a virtual chess and card room. In the virtual social scene, the user communicates with other users by controlling a virtual object. The virtual battle scene is a scene in which the user controls the virtual object to battle. The target sub-scene is determined by the first terminal, and may be either the virtual social scene or the virtual battle scene. The type of the sub-scene is not limited in the embodiments of the present disclosure.

[0106] In the above implementation, the user can determine both the first parameter and the target sub-scene for rendering the target virtual object through the first terminal, thereby greatly improving the personalization degree and improving the user viscosity.

[0107] For example, the second server acquires the position of the controlled virtual object in the target virtual scene from the first server, in a case that the first video frame displays the target virtual object of the first terminal. That is, the second server transmits a position acquisition request to the first server. The position acquisition request carries a time stamp of the first video frame and the identifier of the controlled virtual object. After acquiring the position acquisition request, the first server acquires the time stamp of the first video frame and the identifier of the controlled virtual object from the position acquisition request, and performs searching based on the time stamp of the first video frame and the identifier of the controlled virtual object to obtain the position of the controlled virtual object in the target virtual scene at a time point indicated by the time stamp of the first video frame. The first server transmits the position of the controlled virtual object in the target virtual scene to the second server. The position is used for indicating a sub-scene in which the controlled virtual object is located in the target virtual scene. The second server transmits a parameter

acquisition request to the client server while transmitting a position acquisition request to the first server. The parameter acquisition request carries the identifier of the target virtual object. After receiving the parameter acquisition request, the client server acquires the identifier of the target virtual object from the parameter acquisition request. The client server performs searching based on the identifier of the target virtual object to obtain the first parameter corresponding to the target virtual object and the identifier of the target sub-scene, and transmits the first parameter and the identifier of the target sub-scene to the second server. The second server acquires the position transmitted by the first server, the first parameter and the identifier of the target sub-scene transmitted by the client server. The second server renders the target virtual object in the first video frame by using the first parameter to obtain the second video frame in a case that the sub-scene indicated by the position is the target sub-scene. The second server does not render the first video frame, and directly determines the first video frame as the second video frame in a case that the sub-scene indicated by the position is not a target sub-scene, that is to say, the second server can directly transmit the first video frame to the first terminal, and the first terminal displays the first video frame.

[0108] For example, the target sub-scene is a virtual fishing scene, and the target virtual object is a virtual vehicle. When determining that the first video frame displays the virtual vehicle and the controlled virtual object is located in the virtual fishing scene, the second server can perform secondary rendering on the virtual vehicle by using the first parameter to change the display effect of the virtual vehicle. When determining that the first video frame displays the virtual vehicle but the controlled virtual object is not located in the virtual fishing scene, the second server directly determines the first video frame as a second video frame without performing secondary rendering on the first video frame by using the first rendering parameter.

[0109] In step 302 above, in addition to determining the type of the first video frame, the second server can determine the position of the target virtual object in the first video frame in a case that the type of the first video frame is the first type. If the second server determines the type of the first video frame by a template matching method, a scene server can determine the region that matches the template image of the target virtual object in the first video frame as the position of the target virtual object in the first video frame. If the second server determines the type of the first video frame by using the image recognition model, the image recognition model can output a detection box in the first video frame. The detection box indicates the position where the target virtual object is located in the first video frame. After determining the position of the target virtual object in the first video frame, the second server can perform secondary rendering on the position of the target virtual object in the first video frame based on the first rendering parameter to obtain a second video frame.

[0110] For example, the second server transmits a relative position acquisition request to the first server in a case that the first video frame displays the target virtual object of the first terminal. The relative position acquisition request is used for acquiring the distance between the controlled virtual object and the target virtual object. The relative position acquisition request carries the identifier of the controlled virtual object and the identifier of the target virtual object. After receiving the relative position acquisition request, the first server acquires the identifier of the controlled virtual object and the identifier of the target virtual object from the relative position acquisition request, and determines the first angle and the first distance between the controlled virtual object and the target virtual object based on the identifier of the controlled virtual object and the identifier of the target virtual object. The first server transmits the first angle and the first distance between the controlled virtual object and the target virtual object to the second server. After receiving the first angle and the first distance between the controlled virtual object and the target virtual object, the second server transmits a rendering parameter acquisition request to the client server. The rendering parameter acquisition request carries the first angle, the first distance, and the identifier of the target virtual object. After receiving the rendering parameter acquisition request, the client server acquires the first angle, the first distance, and the identifier of the target virtual object from the rendering parameter acquisition request, and determines the first parameter based on the identifier of the target virtual object. The client server searches the first parameter based on the first angle and the first distance to obtain the first rendering parameter, and transmits the first rendering parameter to the second server. The second server determines target pixel values of multiple target pixel points of the target virtual object in the first video frame based on the first rendering parameter. The second server updates the pixel values of the multiple target pixel points in the first video frame by using the target pixel values to obtain the second video frame, that is, replaces the pixel values of the multiple target pixel points with the target pixel values. In some embodiments, the target pixel values are also referred to as texture data.

[0111] With reference to FIG. 4, the first server renders the target virtual scene from the perspective of the controlled virtual object in the target virtual scene to obtain a first video frame 401. The first video frame 401 includes a virtual vehicle 402. The virtual vehicle 402 is also a target virtual object. The second server performs secondary rendering on the virtual vehicle 402 by using the first rendering parameter to obtain a second video frame 403. It can be seen that the virtual vehicle 404 in the second video frame 403 has a different display effect from that in the first video frame 401.

[0112] In some embodiments, there is an Android container in the first server. Rendering is performed inside the Android container. The principle thereof is that a graphics processing unit (GPU) may be directly accessed inside the Android container. The GPU has high concurrent computing

power, and can complete rendering directly inside the Android container. The rendering performance can be improved, and the delay caused by excessive rendering is greatly reduced. One Android container is one complete Android system, which also has perfect compatibility, and can realize rendering of games. When a game starts, the first server can transmit a rendering-completed instruction to the client server. The client server can record game duration information of a current user account in real time. If the duration is insufficient, the client server may instruct a client to process through signaling (for example, when the duration is insufficient, a box pops up to prompt that the duration is not sufficient; when a cloud game is abnormal, the client pops up a box to prompt the user for an abnormality), and the like.

[0113] A method for generating a first parameter of a target virtual object by a client server is described below.

[0114] In one possible implementation, the first terminal displays a configuration interface of the target virtual object. The configuration interface is used for acquiring configuration information of the target virtual object. The configuration information of the target virtual object is transmitted to the client server in response to an operation on the configuration interface. The client server receives the configuration information of the target virtual object, creates a three-dimensional model of the target virtual object based on the configuration information of the target virtual object, and acquires a first parameter of the three-dimensional model. In some embodiments, the client server can bind and store the user account (User ID) logged in by the first terminal, an identifier of the target virtual scene (Game ID), an identifier of the target virtual object (Object ID), and the first parameter, so that the first parameter can be determined based on the identifiers in a subsequent calling process, thereby improving the processing efficiency.

[0115] For example, the first terminal displays a configuration interface of a target cloud game. The configuration interface displays multiple virtual objects in the target cloud game. The target cloud game is a cloud game selected by the user. The game scene of the target cloud game is also a target virtual scene. In response to the target virtual object of the multiple virtual objects being selected, the first terminal displays a configuration interface of the target virtual object. The configuration interface displays a multiple configuration options. The first terminal generates configuration information of the target virtual object based on a selected option of the multiple configuration options, and transmits the configuration information of the target virtual object to the client server. After receiving the configuration information of the target virtual object, the client server renders an initial three-dimensional model of the target virtual object based on the configuration information of the target virtual object to obtain the three-dimensional model of the

target virtual object. The client server acquires rendering parameters of the virtual camera and the three-dimensional model of the target virtual object at different angles and different distances. The rendering parameters at different angles and different distances constitute a first parameter of the three-dimensional model.

[0116] For example, a first terminal starts a cloud game client, and displays a running interface of the cloud game client. The running interface displays multiple cloud games to be selected. In response to the target cloud game of the multiple cloud games being selected, the first terminal displays a configuration interface of the target cloud game. The configuration interface displays multiple virtual objects that may be subjected to personalization processing in the target cloud game, for example, the multiple virtual objects include a virtual vehicle, a virtual tree, or a virtual house. It is assumed that the target virtual object is the virtual vehicle. In response to the target virtual object of the multiple virtual objects being selected, the first terminal displays a configuration interface 501 of the virtual vehicle, as shown in FIG. 5. The configuration interface 501 displays multiple first-level configuration options 502. The first-level configuration options are used for selecting a type of the virtual vehicle. In some embodiments, the first terminal represents the type of the virtual vehicle by using a brand of a vehicle. In response to any one of the multiple first-level configuration options being selected, the first terminal displays multiple second-level configuration options 503 corresponding to the first-level configuration options. The second-level configuration options are used for selecting a style of the virtual vehicle. In some embodiments, the first terminal represents the style of the virtual vehicle by using a model of a vehicle. In response to any one of the multiple secondary configuration options being selected, the first terminal displays multiple third-level configuration options 504 corresponding to the second-level configuration options. The third-level configuration options are used for selecting a color of the virtual vehicle, such as red, black, or blue. In some embodiments, in response to any one of multiple three-level configuration options being selected, the first terminal displays multiple four-level configuration options 505. The four-level configuration options are used for selecting an application range of the configuration information of the target virtual object. The application range includes "visible only to yourself", "visible to all", "permanent use", and "single use". The "visible only to yourself" is used for instructing the second server to perform secondary rendering only on a video frame corresponding to the first terminal and not to perform secondary rendering on video frames corresponding to other terminals during gaming. The "visible to all" is used for indicating that the second server performs secondary rendering on the video frames corresponding to all terminals during gaming. For example, the virtual vehicle in the target virtual scene is displayed as type A-style B-red by default. The user of the first terminal sets the virtual vehicle as type C-style D-white through the configuration interface of the virtual vehicle.

If the user of the first terminal selects the "visible only to yourself", the virtual vehicle is displayed as type C-style D-white in the target virtual scene displayed by the first terminal during gaming. For other terminals displaying the target virtual scene, the virtual vehicle is still displayed as type A-style B-red. Certainly, if the user of the first terminal selects the "visible to all", all terminals displaying the target virtual scene display the virtual vehicle as type C-style D-white during gaming. For the option "permanent use", the second server determines rendering parameters each time according to the configuration information determined this time. For the option "single use", the second server determines rendering parameters only once according to the configuration information determined currently in this gaming process. The first terminal determines a setting parameter of the target virtual object based on the above first-level configuration option, second-level configuration option, third-level configuration option, and fourth-level configuration option, and transmits the setting parameter to the client server. The client server creates a three-dimensional model of the target virtual object by means of an open graphics library (OpenGL) based on the setting parameter of the target virtual object. The client server acquires rendering parameters of the virtual camera and the three-dimensional model of the target virtual object at different angles and different distances. The rendering parameters at different angles and different distances constitute a first parameter of the three-dimensional model.

[0117] In some embodiments, taking the target virtual scene including a virtual vehicle as an example, a technician can create a large number of three-dimensional models of virtual vehicle in advance based on basic information (CarType, CarStyle, CarColor, and the like) of the vehicle through the client server, so that the corresponding three-dimensional model can be directly called during gaming, thereby improving the processing efficiency.

[0118] In an embodiment, after acquiring the second video frame, the second server perform the following steps.

[0119] In one possible implementation, the second server acquires a second animation and second audio corresponding to the target virtual object. The second server adds the second animation corresponding to the target virtual object in the second video frame to obtain a seventh video frame. The second server transmits the seventh video frame and the second audio to the first terminal. The first terminal plays the second audio while displaying the seventh video frame. The position where the second animation is added to the second video frame is determined by a user or a technician. For example, the second animation is added to a position adjacent to the target virtual object, such as a position above or beside the target virtual object. The position is not limited in the embodiments of the present disclosure.

[0120] In this implementation, after performing secondary rendering on the first video frame based on the first parameter to obtain the second video frame, the second server can perform further rendering on the second video frame, that is, add a second animation corresponding to the target virtual object to the second video frame to obtain the seventh video frame. In addition, the second audio corresponding to the target virtual object is also acquired. After the seventh video frame and the second audio are simultaneously transmitted to the first terminal, the first terminal can play the second audio while displaying the seventh video frame. Since the first parameter, the second animation, and the second audio are all determined by the user through the first terminal, the same first video frame may be rendered into different seventh video frames by the second server through different settings, and different second audio is configured for the different seventh video frames, thereby improving the personalization.

[0121] For example, the second server acquires the second animation and the second audio corresponding to the target virtual object from the client server. Both the second animation and the second audio are determined by the first terminal. That is, the second server transmits a first animation acquisition request to the client server. The first animation acquisition request carries the identifier of the target virtual object. After receiving the first animation acquisition request, the client server acquires the identifier of the target virtual object from the first animation acquisition request, performs searching based on the identifier of the target virtual object to acquire the second animation and the second audio corresponding to the target virtual object, and transmits the second animation and the second audio to the second server. After receiving the second animation and the second audio, the second server adds the second animation to the second video frame, for example, adding a first frame of the second animation to the second video frame to obtain the seventh video frame. The second server transmits the seventh video frame and the second audio to the first terminal. The first terminal receives the seventh video frame and the second audio, and plays the second audio while displaying the seventh video frame.

[0122] From a perspective of the user, if the target virtual object is a virtual vehicle, the second animation is a thumb gradually appearing, and the first parameter instructs adjusting a color of the virtual vehicle to red. Then, in the seventh video frame displayed by the first terminal, the color of the virtual vehicle is red, the thumb is displayed beside the virtual vehicle, and the second audio is played simultaneously.

[0123] In step 304, the second server transmits the first video frame to the first terminal in a case that the type indicates that the first video frame does not display the target virtual object of the first terminal. The first terminal is configured to display the first video frame.

[0124] Through step 304, when the first video frame does not display the target virtual object, the second server is unnecessary to perform secondary rendering on the first video frame and directly forwards the first video frame to the first terminal, and the first terminal displays the first video frame.

[0125] In step 305, the second server transmits the second video frame to the first terminal. The first terminal is configured to display the second video frame.

[0126] Through step 305, in a case that the first video frame displays the target virtual object, the second server performs secondary rendering on the first video frame based on the first rendering parameter to obtain a second video frame, and transmits the second video frame to the first terminal. The target virtual object in the second video frame displayed on the first terminal is displayed with a display effect configured by the first terminal, thereby achieving quick adjustment of the display effect of the target virtual object.

[0127] In an embodiment, after transmitting the second video frame to the first terminal, the second server perform any one of the following steps.

[0128] In one possible implementation, the second server aggregates multiple second video frames into a first video frame set, and transmits the first video frame set to the first terminal. The first terminal is configured to share the first video frame set with other terminals.

[0129] In this implementation, since the second video frame is a video frame obtained by performing secondary rendering by the second server based on user settings, the display effect of the second video frame is personalized. The second server can aggregate the second video frames into the first video frame set, and transmit the first video frame set to the first terminal. The first terminal can share the first video frame set with other terminals, that is, the user shares his/her personalization for the cloud application with other users to stimulate other users to play the cloud application, so that the cloud application is propagated more widely.

[0130] In some embodiments, after acquiring the first video frame set, the first terminal can share the first video frame set through a social network, so that the first video frame set is propagated more widely.

[0131] In one possible implementation, the second server stitches the second video frame with the first video frame to obtain a stitched video frame. The second server aggregates multiple stitched video frames into a second video frame set, and transmits the second video frame set to the first terminal. The first terminal is configured to share the second video frame set with other terminals.

[0132] In this implementation, the second video frame is a video frame obtained by performing secondary rendering by the second server based on user settings. The first video frame is

a video frame rendered by the first server by default. The user can see the personalization of the user more clearly by viewing the stitched video frame after the second video frame and the first video frame are stitched into the stitched video frame. The second server can aggregate the stitched video frames into a second video frame set, and transmit the second video frame set to the first terminal. The first terminal can share the second video frame set with other terminals, that is, share his/her personalization for a cloud application with other users to stimulate other users to play the cloud application, so that the cloud application is propagated more widely.

[0133] In the above steps 301 to 305, the description is made by assuming that the second server performs processing on the first video frame to obtain the second video frame. During gaming, the first server performs real-time rendering on the target scene to obtain a series of video frames. The series of video frames all correspond to the first terminal. The first video frame is one video frame in the series of video frames. For each of the series of video frames, the second server can perform steps 301 to 305 above.

[0134] In an embodiment, if the first terminal selects the "visible to all", the second server can also perform the following steps 306 to 308, so that the second server performs secondary rendering on video frames corresponding to other terminals to change the display effect of the target virtual object in the virtual scene.

[0135] In step 306, the second server acquires a fifth video frame corresponding to the second terminal. The fifth video frame is a video frame obtained by rendering the target virtual scene from a perspective of a second virtual object.

[0136] A method for acquiring the fifth video frame corresponding to the second terminal by the second server share the same concept as the method for acquiring the first video frame corresponding to the first terminal by the second server. For an implementation process, one may refer to the relevant description of the above step 301, and details are not described here.

[0137] In step 307, the second server performs target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object.

[0138] A method for determining whether the fifth video frame displays the target virtual object by the second server share the same concept as the method for determining whether the first video frame displays the target virtual object by the second server. For an implementation process, one may refer to the relevant description of the above step 302, and details are not described here.

[0139] In one possible implementation, the second server renders the target virtual object in the fifth video frame based on a second parameter of the second terminal to obtain a sixth video frame

in a case that the fifth video frame displays the target virtual object. In an embodiment, the second parameter is determined by the second terminal.

[0140] In this case, if the second terminal sets the second parameter for the target virtual object, the second server can render the target virtual object in the fifth video frame by using the second parameter to obtain the sixth video frame, and can transmit the sixth video frame to the second terminal subsequently for displaying by the second terminal. That is, the second server can perform secondary rendering on the video frame including the target virtual object by using the first parameter, and transmit the video frame subjected to the secondary rendering to the first terminal. The second server performs secondary rendering on the video frame including the target virtual object by using the second parameter, and transmit the video frame subjected to the secondary rendering to the second terminal. The target virtual object may have different display effects in the video frames displayed on the first terminal and the second terminal. For example, taking the target virtual object being a virtual vehicle as an example, if the first parameter indicates that the virtual vehicle is rendered to red and the second parameter indicates that the virtual vehicle is rendered to blue, the virtual vehicle is red in the video frame displayed by the first terminal, and the virtual vehicle is blue in the video frame displayed by the second terminal.

[0141] For example, if the second terminal determines the second parameter for the target virtual object, the second server acquires a second angle and a second distance between the second virtual object and the target virtual object from the first server in a case that the fifth video frame displays the target virtual object. The second server transmits the second angle, the second distance, and the identifier of the target virtual object to the client server. After receiving the second angle, the second distance, and the identifier of the target virtual object, the client server determines the second parameter based on the identifier of the target virtual object. The client server searches the second parameter based on the second angle and the second distance to obtain a third rendering parameter, and transmits the third rendering parameter to the second server. After receiving the third rendering parameter, the second server renders the target virtual object in the fifth video frame by using the third rendering parameter to obtain a sixth video frame.

[0142] In one possible implementation, the second server renders the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame, in a case that the fifth video frame displays the target virtual object.

[0143] In this case, if the second terminal does not set the second parameter for the target virtual object, the second server can render the target virtual object in the fifth video frame by using the first parameter to obtain the sixth video frame, and can transmit the sixth video frame to the second

terminal subsequently for displaying by the second terminal. That is, the target virtual object has the same display effect in the video frames displayed on the first terminal and the second terminal.

[0144] For example, if the second terminal does not determine the second parameter for the target virtual object, the server acquires the second angle and the second distance between the second virtual object and the target virtual object from the first server in a case that the fifth video frame displays the target virtual object. The second server transmits the second angle, the second distance, and the identifier of the target virtual object to the client server. After receiving the second angle, the second distance, and the identifier of the target virtual object, the client server determines the first parameter based on the identifier of the target virtual object. The client server searches the first parameter based on the second angle and the second distance to obtain a third rendering parameter, and transmits the third rendering parameter to the second server. After receiving the third rendering parameter, the second server renders the target virtual object in the fifth video frame by using the third rendering parameter to obtain a sixth video frame.

[0145] In one possible implementation, the second server adds a second animation corresponding to the target virtual object to the fifth video frame to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object.

[0146] In this implementation, the second server can add a second animation corresponding to the target virtual object to the fifth video frame, thereby improving the personalization.

[0147] In one possible implementation, both the controlled virtual object and the second virtual object are located in a target sub-scene of the target virtual scene, and the second server compares virtual levels of the controlled virtual object and the second virtual object in a case that the fifth video frame displays the target virtual object. The second server renders the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is higher than that of the second virtual object. The second server renders the target virtual object in the fifth video frame based on the second parameter of the second terminal to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is lower than that of the second virtual object. The second server renders the target virtual object in the fifth video frame based on the third parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is the same as the virtual level of the second virtual object. In an embodiment, the third parameter is set by the first server.

[0148] The target virtual scene includes multiple sub-scenes, and the multiple sub-scenes constitute the target virtual scene. For example, the multiple sub-scenes include a virtual social scene and a virtual battle scene. The virtual social scene includes a virtual fishing scene, a virtual chat room,

a virtual dance room, or a virtual chess and card room. In the virtual social scene, the user communicates with other users by controlling a virtual object. The virtual battle scene is a scene in which the user controls the virtual object to battle. The target sub-scene is determined by the first terminal, and may be either the virtual social scene or the virtual battle scene. The target sub-scene is not limited in the embodiments of the present disclosure. The virtual level is a level of the virtual object in the target virtual scene. A high virtual indicates a strong battle capability of the virtual object. In an embodiment, the virtual level is a member level of a game account corresponding to the virtual object. A higher member level indicates more services can be enjoyed in a cloud application.

[0149] In this implementation, when both the controlled virtual object and the second virtual object are located in the target sub-scene, the second server can determine, based on the virtual levels of the controlled virtual object and the second virtual object, to perform secondary rendering on the target virtual object by using the first parameter set by the first terminal or the second parameter set by the second terminal, thereby encouraging the user to improve the virtual level of the virtual object and thus improving the game enthusiasm of the user.

[0150] For example, the second server transmits a virtual level acquisition request to the client server in a case that both the controlled virtual object and the second virtual object are located in a target sub-scene of the target virtual scene. The virtual level acquisition request carries the identifier of the controlled virtual object and the identifier of the second virtual object. After acquiring the virtual level acquisition request, the client server acquires the identifier of the controlled virtual object and the identifier of the second virtual object from the virtual level acquisition request, and performs searching based on the identifier of the controlled virtual object and the identifier of the second virtual object to obtain a first virtual level of the controlled virtual object and a second virtual level of the second virtual object. The client server transmits the first virtual level of the controlled virtual object and the second virtual level of the second virtual object to the second server. After the second server receives the first virtual level of the controlled virtual object and the second virtual level of the second virtual object, the second server acquires the first parameter from the client server and renders the target virtual object in the fifth video frame by using the first parameter to obtain the sixth video frame in a case that the first virtual level is higher than the second virtual level. The second server acquires the second parameter from the client server and renders the target virtual object in the fifth video frame by using the second parameter to obtain the sixth video frame in a case that the first virtual level is lower than the second virtual level. The second server acquires the third parameter from the client server, and renders the target virtual object in the fifth video frame by using the third parameter to obtain the sixth video frame in a case that the first virtual level is equal to the second virtual level. In some embodiments, the third parameter is determined based on the first virtual level.

[0151] In step 308, the second server transmits the sixth video frame to the second terminal. The second terminal is configured to display the sixth video frame.

[0152] An optional embodiment of the present disclosure may be formed by using any combination of all the foregoing optional technical solutions, and details are not described herein.

[0153] In the above embodiments of the present disclosure, the method is performed by the second server as an example. In other embodiments, the method may be performed by other servers.

[0154] According to the technical solutions provided by the embodiments of the present disclosure, if it is desired to change a display effect of a certain virtual object, the first parameter is determined for the virtual object by the first terminal, and the server performs secondary rendering on the first video frame based on the first parameter to obtain the second video frame. Compared with the first video frame, the display effect of the virtual object displayed in the second video frame is also the display effect configured by the first terminal for the virtual object. According to such technical solutions, a function of performing secondary rendering on the virtual object is achieved in a cloud application, and the user can quickly and efficiently adjust the display effect of the certain virtual object in the virtual scene, thereby extending a functional range of the cloud application and improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

[0155] In addition to the above steps 301 to 308, another video frame rendering method is further provided according to an embodiment of the present disclosure. The video frame rendering method is applicable to a case that a target event occurs in a target virtual scene. With reference to FIG. 6, the method includes the following steps 601 to 603.

[0156] In step 601, a second server acquires a third video frame corresponding to the first terminal. The third video frame is a video frame obtained by rendering the target virtual scene from a perspective of the controlled virtual object after the target event occurs in the target virtual scene, and the target event is an event associated with the controlled virtual object.

[0157] In a possible implementation, in response to the target event occurring in the target virtual scene, the first server transmits the third video frame corresponding to the first terminal to the second server. The first server transmits prompt information bound to the third video frame to the second server while transmitting the third video frame to the second server. The prompt information carries an identifier of the target event. The second server can determine that the target event occurs in the target virtual scene based on the prompt information while acquiring the third video frame, so as to trigger subsequent rendering of the third video frame.

[0158] For example, the target event is that the controlled virtual object defeats the first virtual object in the target virtual scene. Defeat here means that a life value of the first virtual object is reduced to 0 due to a behavior of the controlled virtual object in the target virtual scene. For example, the controlled virtual object attacks the first virtual object by using a virtual gun/a virtual dagger/a virtual hand grenade, and the like to reduce the life value of the first virtual object to 0. The first virtual object is an object in a different team from the controlled virtual object, or the first virtual object is a virtual object that is hostile to the controlled virtual object. In response to the controlled virtual object defeating the first virtual object in the target virtual scene, the first server transmits the third video frame to the second server in real time. The third video frame is a first video frame after the first virtual object is defeated. The first server transmits prompt information bound to the third video frame to the second server while transmitting the third video frame to the second server. The prompt information, in addition to carrying the identifier of the target event, also carries the position where the first virtual object is defeated. Subsequently, the second server can also determine a region where the first virtual object is defeated in the third video frame, in addition to determining that the controlled virtual object defeats the first virtual object in the target virtual scene based on the prompt information. In some embodiments, in response to the target event occurring in the target virtual scene, the first server can Hook the target event by using a Hook function, and notify the second server of the target event, that is, notify the second server that the target event occurs in the target virtual scene by the prompt information.

[0159] The above example is described by assuming that the target event is that the controlled virtual object defeats the first virtual object in the target virtual scene. In other possible implementations, the target event may be that the controlled virtual object picks up a target virtual prop in the target virtual scene, the controlled virtual object starts a target virtual carrier in the target virtual scene, or the controlled virtual object successively defeat multiple virtual objects in the target virtual scene, and the like. The target event is set by a technician according to actual situations. The type of the target event is not limited in the embodiments of the present disclosure.

[0160] In step 602, the second server adds a first animation corresponding to the target event to the third video frame to obtain a fourth video frame.

[0161] In a possible implementation, the target event is that the controlled virtual object defeats the first virtual object in the target virtual scene, and the second server determines a region where the first virtual object is defeated in the third video frame. The second server adds a first animation corresponding to the target event to the region to obtain the fourth video frame. A correspondence between the target event and the animation is set through the first terminal. For

example, the target event is selected by the first terminal, and the animation bound to the target event is selected from multiple animations provided by the client server; or the first animation corresponding to the target event is uploaded to the client server by the first terminal, and the client server binds the animation to the target event after receiving the animation uploaded by the first terminal.

[0162] For example, the first server transmits the prompt information bound to the third video frame to the second server while transmitting the third video frame to the second server. The second server acquires the identifier of the target event and the position where the first virtual object is defeated from the prompt information. The second server determines the first animation corresponding to the target event based on the identifier of the target event, and determines the region where the first virtual object is defeated based on the position where the first virtual object is defeated. The second server adds the first animation corresponding to the target event to the region to obtain the fourth video frame.

[0163] For example, the first server transmits the prompt information bound to the third video frame to the second server while transmitting the third video frame to the second server. The second server acquires the identifier of the target event and the position where the first virtual object is defeated from the prompt information. The second server transmits a second animation acquisition request to the client server. The second animation acquisition request carries the identifier of the target event. After receiving the second animation acquisition request, the client server acquires the identifier of the target event from the second animation acquisition request, performs searching based on the identifier of the target event to obtain the first animation corresponding to the target event, and transmits the first animation corresponding to the target event to the second server. After receiving the first animation corresponding to the target event, the second server adds the animation to the region where the first virtual object is defeated to obtain the fourth video frame. In some embodiments, the second server can add the first frame of the animation to the third video frame to obtain the fourth video frame.

[0164] In step 603, the second server transmits the fourth video frame to the first terminal. The first terminal is configured to display the fourth video frame.

[0165] In an embodiment, before step 603, the second server can acquire first audio corresponding to the target event. The first audio is audio of the first terminal. The second server transmits the first audio to the first terminal, and the first terminal is configured to play the first audio while displaying the fourth video frame. A correspondence between the target event and the first audio is set through the first terminal. For example, the target event is selected by the first terminal,

and the first audio bound to the target event is selected from multiple pieces of first audio provided by the client server; or the first audio corresponding to the target event is uploaded to the client server by the first terminal, and the client server binds the first audio to the target event after receiving the first audio uploaded by the first terminal.

[0166] For example, the first server transmits the prompt information bound to the third video frame to the second server while transmitting the third video frame to the second server. The second server acquires the identifier of the target event from the prompt information. The second server transmits an animation and audio acquisition request to the client server. The animation and audio acquisition request carries the identifier of the target event. After receiving the second animation acquisition request, the client server acquires the identifier of the target event from the second animation acquisition request, performs searching based on the identifier of the target event to obtain the first animation and the first audio corresponding to the target event, and transmits the first animation and the first audio corresponding to the target event to the second server. After receiving the first animation and the first audio corresponding to the target event, the second server adds the animation to the third video frame to obtain the fourth video frame. The second server transmits the first audio to the first terminal while transmitting the fourth video frame to the first terminal. After receiving the fourth video frame and the first audio, the first terminal plays the first audio while displaying the fourth video frame.

[0167] It is still assumed that the target event is that the controlled virtual object defeats the first virtual object in the virtual scene. In response to the controlled virtual object defeating the first virtual object in the virtual scene, the first server transmits the third video frame and the prompt information bound to the third video frame to the second server. The prompt information carries the identifier of the target event and the position where the first virtual object is defeated. The second server acquires the identifier of the target event and the position where the first virtual object is defeated from the prompt information. The second server transmits an animation and audio acquisition request to the client server. The animation and audio acquisition request carries the identifier of the target event. After receiving the animation and audio acquisition request, the client server acquires the identifier of the target event from the animation and audio acquisition request, performs searching based on the identifier of the target event to obtain the first animation and the first audio corresponding to the target event, and transmits the first animation and the first audio corresponding to the target event to the second server. After receiving the first animation and the first audio corresponding to the target event, the second server adds the animation to the position where the first virtual object is defeated in the third video frame to obtain the fourth video frame. The second server transmits the first audio to the first terminal while transmitting the fourth video frame to the

first terminal. After receiving the fourth video frame and the first audio, the first terminal plays the first audio while displaying the fourth video frame. If the animation is an animation that a small person dances, the first audio is played while displaying the animation that the small person dances at the position where the first virtual object is defeated in the fourth video frame, and the user stimulates himself to achieve better performance in a game by means of the animation and the first audio. With reference to FIG. 7, the second server adds an animation 703 at a position 702 where the first virtual object is defeated in a fourth video frame 701.

[0168] According to the technical solution provided by the embodiments of the present disclosure, if it is desired to play a specified animation while transmitting the target event in the target virtual scene, a target animation is selected for the target event by the first terminal. The second server performs secondary rendering on the video frame to obtain the fourth video frame. Compared with the third video frame, the animation set by the first terminal can be displayed in the fourth video frame. According to such technical solutions, a function of performing secondary rendering on the animation is achieved in a cloud application, and the user can quickly and efficiently adjust the display effect of the virtual scene, thereby extending a functional range of the cloud application and improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

[0169] In the above description process, the first terminal, the client server, the first server, the second server, and the game server constitute one video frame rendering system. An implementation environment of the video frame rendering method is introduced in FIG. 1, and the functions of various components are briefly introduced above. A video frame rendering system provided by embodiments of the present disclosure is introduced below in combination with the above method embodiments. With reference to FIG. 1, the system includes: a first terminal, a first server, and a second server. The first terminal, the first server, and the second server are in communication connection with each other.

[0170] In an example, the first server is a cloud server.

[0171] In an example, the second server is a scene management server.

[0172] The first server is configured to render a target virtual scene from a perspective of a controlled virtual object in the target virtual scene to obtain a first video frame corresponding to the first terminal, and transmit the first video frame to the second server. The controlled virtual object is a virtual object controlled by the first terminal.

[0173] The second server is configured to receive the first video frame.

[0174] The second server is further configured to render a target virtual object in the first video frame based on a first parameter of the first terminal to obtain a second video frame, in a case that the first video frame displays the target virtual object of the first terminal.

[0175] The second server is further configured to transmit the second video frame to the first terminal.

[0176] The first terminal is configured to display the second video frame in response to receiving the second video frame.

[0177] In a possible implementation, the first terminal is further configured to transmit control information to the first server. The control information is used for controlling an action of the controlled virtual object in the target virtual scene.

[0178] The first server is further configured to determine a perspective of the controlled virtual object in the target virtual scene based on the control information, and render the target virtual scene from the perspective of the controlled virtual object in the target virtual scene to obtain the first video frame.

[0179] In a possible implementation, the system further includes a client server. The client server is in communication connection with each of the first terminal, the first server, and the second server.

[0180] The first terminal is further configured to display a configuration interface of the target virtual object. The configuration interface is used for acquiring configuration information of the target virtual object.

[0181] The first terminal is further configured to transmit the configuration information of the target virtual object to the client server in response to an operation on the configuration interface.

[0182] The client server is configured to receive the configuration information of the target virtual object, create a three-dimensional model of the target virtual object based on the configuration information of the target virtual object, and acquire a first parameter of the three-dimensional model.

[0183] In a possible implementation, the second server is further configured to acquire a first rendering parameter in a case that the first video frame displays a target virtual object of the first terminal. The first rendering parameter is a rendering parameter corresponding to a first angle and a first distance in the first parameter. The first angle is an angle between the controlled virtual object and the target virtual object. The first distance is a distance between the controlled virtual object and the target virtual object. The target virtual object in the first video frame is rendered based on the first rendering parameter to obtain the second video frame.

[0184] In a possible implementation, the second server is further configured to determine target pixel values of multiple target pixel points of the target virtual object in the first video frame based on the first rendering parameter. The pixel values of the multiple target pixel points in the first video frame are updated by using the target pixel values to obtain the second video frame.

[0185] In a possible implementation, the second server is further configured to acquire a first rendering parameter and a second rendering parameter from the first parameter in a case that the first video frame displays the target virtual object of the first terminal. In the first video frame, the target virtual object and the controlled virtual object are rendered respectively by using the first rendering parameter and the second rendering parameter to obtain the second video frame.

[0186] In a possible implementation, the second server is further configured to determine a position of the controlled virtual object in the target virtual scene in a case that the first video frame displays the target virtual object of the first terminal. The target virtual object in the first video frame is rendered based on the first parameter to obtain the second video frame in a case that the controlled virtual object is located in a target sub-scene of the target virtual scene. The first video frame is determined as the second video frame in a case that the controlled virtual object is not located in the target sub-scene of the target virtual scene.

[0187] In a possible implementation, the second server is further configured to perform image recognition on the first video frame to determine a type of the first video frame. The target virtual object is rendered based on the first parameter to obtain the second video frame in a case that the type indicates that the first video frame displays the target virtual object.

[0188] In a possible implementation, the second server is further configured to perform detection in the first video frame by using a template image of the target virtual object. In response to detecting that there is a region that matches the template image of the target virtual object in the first video frame, a type of the first video frame is determined as a first type. The first type indicates that the first video frame displays the target virtual object. In response to detecting that there is no region that matches the template image of the target virtual object in the first video frame, the type of the first video frame is determined as a second type. The second type indicates that the first video frame does not display the target virtual object.

[0189] In a possible implementation, the second server is further configured to input the first video frame into an image recognition model. Feature extraction and classification are performed on the first video frame through the image recognition model, to output the type of the first video frame.

[0190] In a possible implementation, the second server is further configured to divide the first video frame into multiple image blocks, and input the multiple image blocks into the image recognition model. The process of performing the feature extraction and classification on the first video frame through the image recognition model, to output the type of the first video frame includes: performing feature extraction and full connection processing on the multiple image blocks by the image recognition model to obtain multiple probabilities respectively corresponding to the multiple image blocks, the probabilities being the probabilities that the corresponding image blocks include the target virtual object. In response to any of the multiple probabilities being greater than or equal to a probability threshold, the type of the first video frame is determined as the first type. The first type indicates that the first video frame displays the target virtual object. In response to each of the multiple probabilities being less than the probability threshold, the type of the first video frame is determined as the second type. The second type indicates that the first video frame does not display the target virtual object.

[0191] In a possible implementation, the second server is further configured to acquire the first parameter from the client server. The client server is configured to create a three-dimensional model of the target virtual object based on a setting parameter uploaded by the first terminal for the target virtual object. The first parameter includes multiple rendering parameters of the three-dimensional model.

[0192] In a possible implementation, the first server is further configured to render the target virtual scene from a perspective of the controlled virtual object to obtain a third video frame corresponding to the first terminal in response to a target event occurring in the target virtual scene, and transmit the third video frame to the second server. The target event is that the controlled virtual object defeats the first virtual object in the target virtual scene.

[0193] The second server is further configured to acquire a first animation corresponding to the target event and first audio corresponding to the target event. The first audio is audio of the first terminal.

[0194] The second server is further configured to determine a region where the first virtual object is defeated in the third video frame. A first animation corresponding to the target event is added to the region to obtain a fourth video frame. The fourth video frame and the first audio are transmitted to the first terminal.

[0195] The first terminal is further configured to play the first audio while displaying the fourth video frame in response to receiving the fourth video frame and the first audio.

[0196] In a possible implementation, the system further includes a second terminal. The second terminal, the first server, and the second server are in communication connection with each other. The second terminal and the first terminal are different terminals.

[0197] The first server is further configured to render the target virtual scene from a perspective of a second virtual object in the target virtual scene to obtain a fifth video frame corresponding to the second terminal, and transmit the fifth video frame to the second server. The second virtual object is a virtual object controlled by the second terminal.

[0198] The second server is further configured to perform target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object, and transmit the sixth video frame to the second terminal.

[0199] The second terminal is configured to display the sixth video frame in response to receiving the sixth video frame.

[0200] In a possible implementation, the second server is further configured to perform any one of operations of:

[0201] rendering the target virtual object in the fifth video frame based on the second parameter of the second terminal to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object;

[0202] rendering the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object; and

[0203] adding a second animation corresponding to the target virtual object to the fifth video frame to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object.

[0204] In a possible implementation, the second server is further configured to compare the virtual levels of the controlled virtual object and the second virtual object in a case that the fifth video frame displays the target virtual object. The target virtual object in the fifth video frame is rendered based on the first parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is higher than the virtual level of the second virtual object. The target virtual object in the fifth video frame is rendered based on the second parameter of the second terminal to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is lower than the virtual level of the second virtual object. The target virtual object in the fifth video frame is

rendered based on a third parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is the same as the virtual level of the second virtual object.

[0205] In a possible implementation, the second server is further configured to acquire a second animation and second audio corresponding to the target virtual object. The second animation corresponding to the target virtual object is added to the second video frame to obtain a seventh video frame. The seventh video frame and the second audio are transmitted to the first terminal. The first terminal is configured to play the second audio while displaying the seventh video frame.

[0206] In a possible implementation, the second server is further configured to perform any one of operations of:

[0207] aggregating multiple second video frames into a first video frame set, and transmitting the first video frame set to the first terminal, the first terminal being configured to share the first video frame set with other terminals;

[0208] stitching the second video frame and the first video frame to obtain a stitched video frame; and aggregating multiple stitched video frames into a second video frame set, and transmitting the second video frame set to the first terminal, the first terminal being configured to share the second video frame set with other terminals.

[0209] In a possible implementation, the second server is further configured to acquire the first parameter from the client server. The client server is configured to create a three-dimensional model of the target virtual object based on a setting parameter uploaded by the first terminal for the target virtual object. The first parameter includes multiple rendering parameters of the three-dimensional model.

[0210] A video frame rendering system provided by the embodiments of the present disclosure is described below with reference to FIG. 1 and steps 301 to 308 and 601 to 603 above.

[0211] The first terminal displays a configuration interface of the target virtual object. The configuration interface is used for acquiring configuration information of the target virtual object. The configuration information of the target virtual object is transmitted to the client server in response to an operation on the configuration interface. The client server receives the configuration information of the target virtual object, creates a three-dimensional model of the target virtual object based on the configuration information of the target virtual object, and acquires a first parameter of the three-dimensional model. Taking a game scene as an example, the first terminal starts a target cloud game, maintains a long connection with a client server, and transmits a game start instruction to the client server. The game start instruction carries a user account, an identifier of the target cloud game, and

hardware information of the first terminal. After receiving the game start instruction, the client server transmits the game start instruction to the first server. After receiving the game start instruction, the first server acquires the user account, the identifier of the target cloud game, and the hardware information of the first terminal from the game start instruction. The first server initializes the target cloud game based on the hardware information of the first terminal to realize matching between a rendered game picture and the first terminal, and transmits the user account to the game server corresponding to the target cloud game. After receiving the user account, the game server transmits information corresponding to the game account to the first server. The first server starts the target cloud game based on the information corresponding to the game account. In a process of running the target cloud game, a target virtual scene of the target cloud game is displayed, and the target virtual scene displays the controlled virtual object. The first terminal transmits control information of the controlled virtual object to the first server. After receiving the control information, the first server determines a perspective of the controlled virtual object in the target virtual scene based on the control information. The first server renders the target virtual scene from the perspective of the controlled virtual object to obtain a first video frame. The first video frame is also a video frame corresponding to the first terminal. The first server transmits the first video frame to the second server. If the user personalizes a certain virtual object in the target cloud game before the game starts, for example, sets a vehicle in the target cloud game to red, the second server can perform image recognition on the first video frame and determine whether the first video frame includes the vehicle. If the second server determines that the first video frame displays the vehicle, a first parameter for the vehicle is acquired from the client server. The first parameter is determined by the client server based on a parameter set for the vehicle by the user. The second server renders the vehicle in the first video frame based on the first parameter to obtain a second video frame, and transmits the second video frame to the first terminal. The user can view the second video frame through the first terminal. If a technician configures the color of the vehicle to blue in the target cloud game, the color of the vehicle can be adjusted to red set by the user through the above steps to achieve personalized configuration for the vehicle.

[0212] According to the technical solutions provided by the embodiments of the present disclosure, if it is desired to change a display effect of a certain virtual object, the first parameter is determined for the virtual object by the first terminal, and the server performs secondary rendering on the first video frame based on the first parameter to obtain the second video frame. Compared with the first video frame, the display effect of the virtual object displayed in the second video frame is also the display effect configured by the first terminal for the virtual object. According to such technical solutions, a function of performing secondary rendering on the virtual object is achieved in

a cloud application, and the user can quickly and efficiently adjust the display effect of the certain virtual object in the virtual scene, thereby extending a functional range of the cloud application and improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

[0213] FIG. 8 is a schematic structural diagram of a video frame rendering apparatus according to an embodiment of the present disclosure. With reference to FIG. 8, the apparatus includes: a first video frame acquisition module 801, a rendering module 802, and a transmitting module 803.

[0214] The first video frame acquisition module 801 is configured to acquire a first video frame corresponding to a first terminal. The first video frame is a video frame obtained by rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene. The controlled virtual object is a virtual object controlled by the first terminal.

[0215] The rendering module 802 is configured to render the target virtual object in the first video frame based on the first parameter of the first terminal to obtain a second video frame, in a case that the first video frame displays the target virtual object of the first terminal.

[0216] The transmitting module 803 is configured to transmit the second video frame to the first terminal. The first terminal is configured to display the second video frame.

[0217] In a possible implementation, the rendering module 802 is configured to acquire a first rendering parameter in a case that the first video frame displays a target virtual object of the first terminal. The first rendering parameter is a rendering parameter corresponding to a first angle and a first distance in the first parameter. The first angle is an angle between the controlled virtual object and the target virtual object. The first distance is a distance between the controlled virtual object and the target virtual object. The target virtual object in the first video frame is rendered based on the first rendering parameter to obtain the second video frame.

[0218] In a possible implementation, the rendering module 802 is configured to determine target pixel values of multiple target pixel points of the target virtual object in the first video frame based on the first rendering parameter. The pixel values of the multiple target pixel points in the first video frame are updated by using the target pixel values to obtain the second video frame.

[0219] In a possible implementation, the rendering module 802 is configured to acquire a first rendering parameter and a second rendering parameter from the first parameter in a case that the first video frame displays the target virtual object of the first terminal. In the first video frame, the target

virtual object and the controlled virtual object are rendered respectively by using the first rendering parameter and the second rendering parameter to obtain the second video frame.

[0220] In a possible implementation, the rendering module 802 is configured to determine the position of the controlled virtual object in the target virtual scene in a case that the first video frame displays the target virtual object of the first terminal. The target virtual object in the first video frame is rendered based on the first parameter to obtain the second video frame in a case that the controlled virtual object is located in a target sub-scene of the target virtual scene.

[0221] In a possible implementation, the rendering module 802 is further configured to determine the first video frame as the second video frame in a case that the controlled virtual object is not located in a target sub-scene of the target virtual scene.

[0222] In a possible implementation, the apparatus further includes:

[0223] an image recognition module, configured to perform image recognition on the first video frame, to determine a type of the first video frame.

[0224] The rendering module 802 is configured to render the target virtual object based on the first parameter to obtain the second video frame in a case that the type indicates that the first video frame displays the target virtual object.

[0225] In a possible implementation, the image recognition module is configured to perform detection in the first video frame by using a template image of the target virtual object. In response to detecting that there is a region that matches the template image of the target virtual object in the first video frame, a type of the first video frame is determined as a first type. The first type indicates that the first video frame displays the target virtual object. In response to detecting that there is no region that matches the template image of the target virtual object in the first video frame, the type of the first video frame is determined as a second type. The second type indicates that the first video frame does not display the target virtual object.

[0226] In a possible implementation, the image recognition module is configured to input the first video frame into an image recognition model, perform feature extraction and classification on the first video frame through the image recognition model, to output the type of the first video frame.

[0227] In a possible implementation, the image recognition module is configured to divide the first video frame into multiple image blocks, and input the multiple image blocks into the image recognition model. Feature extraction and full connection processing are performed on the multiple image blocks by the image recognition model to obtain multiple probabilities respectively corresponding to the multiple image blocks. The probabilities are the probabilities that the

corresponding image blocks include the target virtual object. In response to any of the multiple probabilities being greater than or equal to a probability threshold, the type of the first video frame is determined as the first type. The first type indicates that the first video frame displays the target virtual object. In response to each of the multiple probabilities being less than the probability threshold, the type of the first video frame is determined as the second type. The second type indicates that the first video frame does not display the target virtual object.

[0228] In a possible implementation, the apparatus further includes:

[0229] a third video frame acquisition module, configured to acquire a third video frame corresponding to the first terminal. The third video frame is a video frame obtained by rendering a target virtual scene from a perspective of the controlled virtual object after a target event occurs in the target virtual scene. The target event is that the controlled virtual object defeats the first virtual object in the target virtual scene.

[0230] The rendering module 802 is further configured to acquire a first animation corresponding to the target event and first audio corresponding to the target event. The first audio is audio of the first terminal. A region where the first virtual object is defeated is determined in the third video frame. A first animation corresponding to the target event is added to the region to obtain a fourth video frame.

[0231] The transmitting module 803 is further configured to transmit the fourth video frame and the first audio to the first terminal. The first terminal is configured to play the first audio while displaying the fourth video frame.

[0232] In a possible implementation, the target virtual scene further includes a second virtual object. The second virtual object is a virtual object controlled by a second terminal. The second terminal and the first terminal are different terminals. The apparatus further includes:

[0233] a fifth video frame acquisition module, configured to acquire a fifth video frame corresponding to the second terminal. The fifth video frame is a video frame obtained by rendering the target virtual scene from the perspective of the second virtual object.

[0234] The rendering module 802 is further configured to perform target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object.

[0235] The transmitting module 803 is further configured to transmit the sixth video frame to the second terminal. The second terminal is configured to display the sixth video frame.

[0236] In a possible implementation, the rendering module 802 is further configured to perform any one of the following operations:

[0237] rendering the target virtual object in the fifth video frame based on a second parameter of the second terminal to obtain the sixth video frame, in a case that the fifth video frame displays the target virtual object;

[0238] rendering the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame, in a case that the fifth video frame displays the target virtual object; and

[0239] adding a second animation corresponding to the target virtual object to the fifth video frame to obtain the sixth video frame, in a case that the fifth video frame displays the target virtual object.

[0240] In a possible implementation, the controlled virtual object and the second virtual object are simultaneously located in a target sub-scene of the target virtual scene. The rendering module 802 is further configured to compare virtual levels of the controlled virtual object and the second virtual object in a case that the fifth video frame displays the target virtual object. The target virtual object in the fifth video frame is rendered based on the first parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is higher than the virtual level of the second virtual object. The target virtual object in the fifth video frame is rendered based on the second parameter of the second terminal to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is lower than the virtual level of the second virtual object. The target virtual object in the fifth video frame is rendered based on a third parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is the same as the virtual level of the second virtual object.

[0241] In a possible implementation, the rendering module 802 is further configured to acquire a second animation and second audio corresponding to the target virtual object. The second animation corresponding to the target virtual object is added to the second video frame to obtain a seventh video frame.

[0242] The transmitting module 803 is further configured to transmit the seventh video frame and the second audio to the first terminal. The first terminal is configured to play the second audio while displaying the seventh video frame.

[0243] In a possible implementation, the apparatus further includes a video frame set generation module. The video frame set generation module is configured to perform any one of the following operations:

[0244] aggregating multiple second video frames into a first video frame set, and transmitting the first video frame set to the first terminal, the first terminal being configured to share the first video frame set with other terminals;

[0245] stitching the second video frame and the first video frame to obtain a stitched video frame; and aggregating multiple stitched video frames into a second video frame set, and transmitting the second video frame set to the first terminal, the first terminal being configured to share the second video frame set with other terminals.

[0246] In a possible implementation, the apparatus further includes:

[0247] a parameter acquisition module, configured to acquire the first parameter from a client server. The client server is configured to create a three-dimensional model of the target virtual object based on a setting parameter uploaded by the first terminal for the target virtual object. The first parameter includes multiple rendering parameters of the three-dimensional model.

[0248] The secondary rendering performed on the video frame by the video frame rendering apparatus provided in the above embodiments is illustrated by the above functional modules of the video frame rendering apparatus. In practical applications, the above functions may be allocated to different functional modules for completing, that is, a server is divided into different functional modules to complete all or a part of the functions described above. In addition, the video frame rendering apparatus and the video frame rendering method provided by the above embodiments share the same concept. For a specific implementation process, one may refer to the method embodiment, and details are not described again herein.

[0249] According to the technical solutions provided by the embodiments of the present disclosure, if it is desired to change a display effect of a certain virtual object, the first parameter is determined for the virtual object by the first terminal, and the server performs secondary rendering on the first video frame based on the first parameter to obtain the second video frame. Compared with the first video frame, the display effect of the virtual object displayed in the second video frame is also the display effect configured by the first terminal for the virtual object. According to such technical solutions, a function of performing secondary rendering on the virtual object is achieved in a cloud application, and the user can quickly and efficiently adjust the display effect of the certain virtual object in the virtual scene, thereby extending a functional range of the cloud application and

improving the personalization of the cloud application. Therefore, the cloud application is propagated more widely.

[0250] The structure of the server is described below.

[0251] FIG. 9 is a schematic structural diagram of a server according to an embodiment of the present disclosure. The server 900 may have different configurations or performances, and may include one or more central processing units (CPU) 901 and one or more memories 902. The one or more memories 902 store at least one computer program. The at least one computer program is loaded and executed by the one or more processors 901 to implement the method provided by each method embodiment. Certainly, the server 900 may also have a wired or wireless network interface, a keyboard, and an input/output interface, and the like, so as to perform input and output. The server 900 may further include other components for implementing device functions, which will not be described in detail herein.

[0252] In an exemplary embodiment, a computer-readable storage medium is further provided, for example, a memory including a computer program. The above computer program may be executed by the processor to perform the video frame rendering method in the above embodiments. For example, the computer-readable storage medium may be a read-only memory (ROM), a random access memory (RAM), a compact disc read-only memory (CD-ROM), a magnetic tape, a floppy disk, an optical data storage device, and the like.

[0253] In an exemplary embodiment, a computer program product or a computer program is provided. The computer program product or the computer program includes a program code. The program code is stored in the computer-readable storage medium. A processor of a computer device reads the program code from the computer-readable storage medium. The processor executes the program code, so that the computer device performs the video frame rendering method. That is, the program code is executed by the processor to implement the video frame rendering method.

[0254] In some embodiments, a computer program involved in the embodiments of the present disclosure may be deployed to be executed on one computer device, or on multiple computer devices located at one position, or on multiple computer devices distributed at multiple positions and interconnected over a communication network. The multiple computer devices distributed at the multiple positions and interconnected over the communication network may form a block chain system.

[0255] A person of ordinary skill in the art may understand that all or some of the steps of the foregoing embodiments may be implemented by using hardware, or may be implemented by a

program instructing relevant hardware. The program may be stored in a computer-readable storage medium. The above storage medium may be a read-only memory, a magnetic disk, an optical disk, or the like.

[0256] The foregoing descriptions are merely optional embodiments of the present disclosure, and are not intended to limit the present disclosure. Any modification, equivalent replacement, or improvement made without departing from the spirit and principle of the present disclosure shall fall within the scope of the protection of the present disclosure.

CLAIMS

What is claimed is:

1. A video frame rendering method, comprising:

acquiring, by a server, a first video frame corresponding to a first terminal, the first video frame being a video frame obtained by another server through rendering a target virtual scene from a perspective of a controlled virtual object in the target virtual scene, and the controlled virtual object being a virtual object controlled by the first terminal;

rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal, the target virtual object being a virtual object to be re-rendered by the first terminal; and

transmitting, by the server, the second video frame to the first terminal, the first terminal being configured to display the second video frame.

2. The method according to claim 1, wherein the rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal comprises:

acquiring, by the server, a first rendering parameter, in response to determining that the first video frame displays the target virtual object of the first terminal, the first rendering parameter being a rendering parameter corresponding to a first angle and a first distance in the first parameter, the first angle being an angle between the controlled virtual object and the target virtual object, and the first distance being a distance between the controlled virtual object and the target virtual object; and

rendering, by the server, the target virtual object in the first video frame based on the first rendering parameter to obtain the second video frame.

3. The method according to claim 2, wherein the rendering, by the server, the target virtual object in the first video frame based on the first rendering parameter to obtain the second video frame comprises:

determining, by the server, target pixel values of a plurality of target pixel points of the target

virtual object in the first video frame based on the first rendering parameter; and

updating, by the server, pixel values of the plurality of target pixel points in the first video frame by using the target pixel values to obtain the second video frame.

4. The method according to claim 1, wherein the rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal comprises:

acquiring, by the server, a first rendering parameter and a second rendering parameter from the first parameter, in response to determining that the first video frame displays the target virtual object of the first terminal; and

rendering, by the server, the target virtual object and the controlled virtual object in the first video frame respectively by using the first rendering parameter and the second rendering parameter to obtain the second video frame.

5. The method according to claim 1, wherein the rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal comprises:

determining, by the server, a position of the controlled virtual object in the target virtual scene , in response to determining that the first video frame displays the target virtual object of the first terminal; and

rendering, by the server, the target virtual object in the first video frame based on the first parameter to obtain the second video frame in a case that the controlled virtual object is located in a target sub-scene of the target virtual scene.

6. The method according to claim 5, further comprising:

determining, by the server, the first video frame as the second video frame in a case that the controlled virtual object is not located in the target sub-scene of the target virtual scene.

7. The method according to claim 1, wherein before the rendering, by the server, a target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal, the method further comprises:

performing, by the server, image recognition on the first video frame, to determine a type of the first video frame;

wherein the rendering, by the server, a target virtual object in the first video frame based on a first parameter to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal comprises:

rendering, by the server, the target virtual object based on the first parameter to obtain the second video frame, in response to determining that the type of the first video frame indicates that the first video frame displays the target virtual object.

8. The method according to claim 1, further comprising:

acquiring, by the server, a third video frame corresponding to the first terminal, the third video frame being a video frame obtained by rendering the target virtual scene from the perspective of the controlled virtual object after a target event occurs in the target virtual scene, and the target event indicating that the controlled virtual object defeats the first virtual object in the target virtual scene;

acquiring, by the server, a first animation corresponding to the target event and first audio corresponding to the target event, the first audio being audio of the first terminal;

determining, by the server, a region where the first virtual object is defeated in the third video frame;

adding, by the server, the first animation corresponding to the target event to the region to obtain a fourth video frame; and

transmitting, by the server, the fourth video frame and the first audio to the first terminal, the first terminal being configured to play the first audio while displaying the fourth video frame.

9. The method according to claim 1, wherein the target virtual scene further comprises a second virtual object, the second virtual object is a virtual object controlled by a second terminal, the second terminal and the first terminal are different terminals, and the method further comprises:

acquiring, by the server, a fifth video frame corresponding to the second terminal, the fifth video frame being a video frame obtained by rendering the target virtual scene from a perspective of the second virtual object;

performing, by the server, target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object; and

transmitting, by the server, the sixth video frame to the second terminal, the second terminal being configured to display the sixth video frame.

10. The method according to claim 9, wherein the performing, by the server, target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object comprises at least one of:

rendering, by the server, the target virtual object in the fifth video frame based on a second parameter of the second terminal to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object;

rendering, by the server, the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object; and

adding, by the server, a second animation corresponding to the target virtual object to the fifth video frame to obtain the sixth video frame in a case that the fifth video frame displays the target virtual object.

11. The method according to claim 9, wherein both the controlled virtual object and the second virtual object are located in a target sub-scene of the target virtual scene; and wherein the performing, by the server, target processing on the fifth video frame to obtain a sixth video frame in a case that the fifth video frame displays the target virtual object comprises:

comparing, by the server, a virtual level of the controlled virtual object with a virtual level of the second virtual object in a case that the fifth video frame displays the target virtual object;

rendering, by the server, the target virtual object in the fifth video frame based on the first parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is higher than the virtual level of the second virtual object;

rendering, by the server, the target virtual object in the fifth video frame based on a second parameter of the second terminal to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is lower than the virtual level of the second virtual object; and

rendering, by the server, the target virtual object in the fifth video frame based on a third parameter to obtain the sixth video frame in a case that the virtual level of the controlled virtual object is the same as the virtual level of the second virtual object.

12. The method according to claim 1, wherein after the rendering, by the server, a target

virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second video frame, in response to determining that the first video frame displays the target virtual object of the first terminal, the method further comprises:

acquiring, by the server, a second animation and second audio corresponding to the target virtual object;

adding, by the server, the second animation corresponding to the target virtual object to the second video frame to obtain a seventh video frame; and

transmitting, by the server, the seventh video frame and the second audio to the first terminal, the second terminal being configured to play the second audio while displaying the seventh video frame.

13. The method according to any one of claims 1 to 12, wherein after the transmitting, by the server, the second video frame to the first terminal, the first terminal being configured to display the second video frame, the method further comprises at least one of:

aggregating, by the server, a plurality of second video frames into a first video frame set, and transmitting the first video frame set to the first terminal, the first terminal being configured to share the first video frame set with other terminals;

stitching, by the server, the second video frame and the first video frame to obtain a stitched video frame; and aggregating, by the server, a plurality of stitched video frames into a second video frame set, and transmitting, by the server, the second video frame set to the first terminal, the first terminal being configured to share the second video frame set with other terminals.

14. A video frame rendering system, comprising:

a first terminal, a first server, and a second server, the first terminal, the first server, and the second server being in communication connection with each other; wherein

the first server is configured to render a target virtual scene from a perspective of a controlled virtual object in the target virtual scene to obtain a first video frame corresponding to the first terminal, and transmit the first video frame to the second server, wherein the controlled virtual object is a virtual object controlled by the first terminal;

the second server is configured to receive the first video frame;

the second server is further configured to render the target virtual object in the first video frame based on a first parameter of the first terminal to update the first video frame as a second

video frame, in response to determining that the first video frame displays the target virtual object of the first terminal;

the second server is further configured to transmit the second video frame to the first terminal;
and

the first terminal is configured to display the second video frame in response to receiving the second video frame.

15. A server, comprising one or more processors and one or more memories, wherein the one or more memories storing at least one computer program, and the computer program being loaded and executed by the one or more processors to implement the video frame rendering method according to any one of claims 1 to 12.

16. A computer-readable storage medium storing at least one computer program, wherein the computer program being loaded and executed by a processor to implement the video frame rendering method according to any one of claims 1 to 12.

17. A computer program product comprising program code, wherein the program code is executed by a processor to implement the video frame rendering method according to any one of claims 1 to 12.