

METHOD AND APPARATUS FOR DETERMINING TARGET VIRTUAL OBJECT, TERMINAL, AND STORAGE MEDIUM

RELATED APPLICATION

[0001] This application claims priority to Chinese Patent Application No. 202010507449.2, entitled "METHOD AND APPARATUS FOR DETERMINING TARGET VIRTUAL OBJECT, TERMINAL, AND STORAGE MEDIUM" and filed with the China National Intellectual Property Administration on June 5, 2020.

FIELD OF THE TECHNOLOGY

[0002] This application relates to the field of computer technologies, and in particular, to a method and apparatus for determining a target virtual object, a terminal, and a storage medium.

BACKGROUND OF THE DISCLOSURE

[0003] With the development of computer technologies and the diversification of terminal functions, increasing mobile phone games occur, where a multiplayer online battle arena (MOBA) game gradually becomes an extremely important game of the mobiles phone games. During a game, a controlled virtual object controlled by a user using a terminal may cause damage to another virtual object in a virtual scene by using a skill, for example, to reduce virtual health points of a virtual object hit by the skill. If the virtual health points of the virtual object are reduced to zero, the virtual object is defeated.

[0004] At present, in a terminal game, because a button used to trigger a skill is usually a virtual button, the user is unlikely to precisely control the skill to hit an intended target virtual object. When the user sets a hit condition to assist in determining a target virtual object to be hit by a skill, a virtual object that can be defeated by the skill may not satisfy the hit condition set by the user. As a result, a target virtual object determined by a terminal is not consistent with a target virtual object intended by the user, leading to low human-computer interaction efficiency.

SUMMARY

[0005] Embodiments of this application provide a method and apparatus for determining a target virtual object, a terminal, and a storage medium, which can make a target virtual object determined by a terminal consistent with a target virtual object intended by a user, thereby

improving the efficiency of human-computer interaction and optimizing user experience. The technical solutions are as follows:

[0006] According to an aspect, a method for determining a target virtual object is provided, including:

[0007] determining a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user;

[0008] obtaining, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets, the defeated condition indicating whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object; and

[0009] obtaining, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and taking the third virtual object as a target virtual object to be hit by the target skill.

[0010] According to an aspect, an apparatus for determining a target virtual object, including:

[0011] a range determining module, configured to determine a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user;

[0012] a first object obtaining module, configured to obtain, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets, the defeated condition indicating whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object; and

[0013] a second object obtaining module, configured to obtain, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and take the third virtual object as a target virtual object to be hit by the target skill.

[0014] According to an aspect, a terminal is provided, including one or more processors and one or more memories, the one or more memories storing at least one piece of program code, the at

least one piece of program code being loaded and executed by the one or more processors to implement the operations performed in the method for determining a target virtual object according to any one of the foregoing possible implementations.

[0015] According to an aspect, a storage medium is provided, storing at least one piece of program code, the at least one piece of program code being loaded and executed by a processor to implement the operations performed in the method for determining a target virtual object according to any one of the foregoing possible implementations.

[0016] According to an aspect, an application program product is provided, storing one or more instructions, the one or more instructions being executed by a processor of a computer device to implement the method for determining a target virtual object.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] To describe the technical solutions in embodiments of this application more clearly, the following briefly describes the accompanying drawings required for describing the embodiments. Apparently, the accompanying drawings in the following description show merely some embodiments of this application, and a person of ordinary skill in the art may still derive other accompanying drawings from these accompanying drawings without creative efforts.

[0018] FIG. 1 is a schematic diagram of an implementation environment of a method for determining a target virtual object according to an embodiment of this application.

[0019] FIG. 2 is a flowchart of a method for determining a target virtual object according to an embodiment of this application.

[0020] FIG. 3 is a flowchart of a method for determining a target virtual object according to an embodiment of this application.

[0021] FIG. 4 is a schematic diagram of a manner of triggering a skill according to an exemplary embodiment of this application.

[0022] FIG. 5 is a schematic diagram of a target search range according to an embodiment of this application.

[0023] FIG. 6 is a schematic diagram of another target search range according to an embodiment of this application.

[0024] FIG. 7 is a schematic flowchart of obtaining a first virtual object that satisfies a defeated condition according to an embodiment of this application.

[0025] FIG. 8 is a schematic flowchart of another method for determining a target virtual object according to an embodiment of this application.

[0026] FIG. 9 is a structural diagram of a class diagram of a search tree according to an embodiment of this application.

[0027] FIG. 10 is a schematic diagram of a game scene according to an embodiment of this application.

[0028] FIG. 11 is a schematic logic diagram of a search tree according to an embodiment of this application.

[0029] FIG. 12 is a schematic structural diagram of an apparatus for determining a target virtual object according to an embodiment of this application.

[0030] FIG. 13 is a schematic structural diagram of a terminal according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0031] To make objectives, technical solutions, and advantages of this application clearer, the following further describes implementations of this application in detail with reference to the accompanying drawings.

[0032] The terms "first", "second", and the like in this application are used for distinguishing between same items or similar items of which effects and functions are basically the same. It is to be understood that, the "first", "second", and "nth" do not have a dependency relationship in logic or time sequence, and a quantity and an execution order thereof are not limited.

[0033] The term "at least one" in this application means one or more and "a plurality of" means two or more. For example, a plurality of first positions means two or more first positions.

[0034] For ease of understanding the technical solutions in the embodiments of this application, some terms involved in the embodiments of this application are explained below.

[0035] Virtual scene: a virtual scene displayed (or provided) when an application program is run on a terminal. The virtual scene may be a simulated environment of a real world, or may be a semi-simulated and semi-fictional virtual environment, or may be an entirely fictional virtual environment. The virtual scene may be any one of a two-dimensional virtual scene, a 2.5-dimensional virtual scene, and a three-dimensional virtual scene. A dimension of the virtual scene is not limited in the embodiments of this application.

[0036] For example, the virtual scene may include the sky, the land, the ocean, or the like. The land may include environmental elements such as the desert and a city. The user may control a virtual object to move in the virtual scene. Optionally, the virtual scene may be further used for a virtual scene battle between at least two virtual objects, and virtual resources available to the at least two virtual objects are provided in the virtual scene.

[0037] Optionally, the virtual scene may include two symmetric regions. Virtual objects that belong to two opposing camps occupy the regions respectively, and a goal of each side is to destroy a target building/stronghold/fort/crystal deep in the opponent's region to win victory. The symmetric regions are, for example, a lower left corner region and an upper right corner region, or a left side intermediate region and a right side intermediate region. In the embodiments of this application, the camp is an object set including a plurality of virtual objects.

[0038] Virtual object: a movable object in a virtual scene. The movable object may be a virtual character, a virtual animal, a cartoon character, or the like, for example, a character, an animal, a plant, an oil drum, a wall, or a stone displayed in a virtual scene. The virtual object may be a virtual image used for representing a user in the virtual scene. The virtual scene may include a plurality of virtual objects, and each virtual object has a shape and a volume in the virtual scene, and occupies some space in the virtual scene.

[0039] Optionally, when the virtual scene is a three-dimensional virtual scene, the virtual object may be a three-dimensional model, and the three-dimensional model may be a three-dimensional character constructed based on three-dimensional human skeleton technology. The same virtual object may show different external appearances by wearing different skins. In some embodiments, the virtual object may be alternatively implemented by using a 2.5-dimensional model or a two-dimensional model. This is not limited in the embodiments of this application.

[0040] Optionally, the virtual object may be a player character controlled through an operation on a client, or may be an artificial intelligence (AI) character set in a virtual scene battle through training, or may be a non-player character (NPC) set in a virtual scene interaction. Optionally, the virtual object may be a virtual character performing sports in the virtual scene. Optionally, a quantity of virtual objects participating in the interaction in the virtual scene may be preset, or may be dynamically determined according to a quantity of clients participating in the interaction.

[0041] NPC object: a virtual object that is not controlled by a user, which, in forms, may include various forms such as a virtual creature, a virtual item, and a virtual ghost. The NPC object

may belong to a user camp participating in contention, or may not belong to any user camp participating in the contention (that is, a neutral virtual object).

[0042] Neutral virtual object: an NPC object that does not belong to any user camp participating in contention, which may be referred to as the "neutral virtual object". The neutral virtual object may perform contention behavior with a virtual object of any user camp participating in the contention. In other words, the neutral virtual object may attack the virtual object of any user camp participating in the contention, or may be attacked by the virtual object of any user camp participating in the contention. For example, the neutral virtual object may include a first virtual object and a second virtual object involved in the embodiments of this application. The neutral virtual object is usually used as a competitive resource, and provides a gain effect for a capturing party. For example, the neutral virtual object may be contended by various user camps participating in the contention, and a camp to which a user who is the first to defeat (or kill) the neutral virtual object belongs may obtain the gain effect provided by the neutral virtual object.

[0043] MOBA game: a game in which several forts are provided in a virtual scene, and users belonging to different camps control virtual objects to battle in the virtual scene, occupy forts or destroy forts belonging to the opposing camp. For example, in the MOBA game, users may be divided into at least two opposing camps, and different virtual teams in the at least two opposing camps occupy respective map regions, and compete against each other using specific victory conditions as goals. The victory conditions include, but are not limited to at least one of occupying forts or destroying forts belonging to the opposing camps, killing virtual objects belonging to the opposing camps, ensure own survivals in a specified scenario and time, seizing a specific resource, and surpassing interactive scores of the opponent within a specified time. For example, in a mobile phone MOBA game, users may be divided into two opposing camps. The virtual objects controlled by the users are scattered in a virtual scene to compete against each other, and the victory condition is to destroy or occupy all enemy forts.

[0044] Optionally, each virtual team includes one or more virtual objects, for example, 1, 2, 3, or 5. According to a quantity of virtual objects in each team participating in a battle arena, the battle arena may be divided into a 1V1 battle, a 2V2 battle, a 3V3 battle, a 5V5 battle, and the like, where "1V1" means "1 versus 1". Details are not described herein.

[0045] Optionally, the MOBA game may take place in rounds, and each round of the battle arena may have the same map or different maps. A duration of one round of the MOBA game is from a moment at which the game starts to a movement at which the victory condition is met.

[0046] In the MOBA game, the user may control a virtual object to fall freely, glide, parachute, or the like in the sky of the virtual scene, or run, jump, crawl, walk in a stooped posture, or the like on the land, or may control a virtual object to swim, float, dive, or the like in the ocean. Herein, the scenes are merely used as examples for description, which are not specifically limited in the embodiments of this application.

[0047] In the MOBA game, the user may further control the virtual object to cast skills to fight with other virtual objects. For example, skill types of the skills may include an attack skill, a defense skill, a healing skill, a carry skill, an execution skill, and the like. Each virtual object may have one or more fixed skills, and different virtual objects generally have different skills, and different skills may produce different effects. For example, if an attack skill cast by a virtual object hits a hostile virtual object, certain damage is caused to the hostile virtual object, which is generally shown as deducting a part of virtual health points of the hostile virtual object. In another example, if a healing skill cast by a virtual object hits a friendly virtual object, certain healing is produced for the friendly virtual object, which is generally shown as restoring a part of virtual health points of the friendly virtual object, and all other types of skills may produce corresponding effects. Details are not described herein again.

[0048] A system architecture involved in this application is described below.

[0049] FIG. 1 is a schematic diagram of an implementation environment of a method for determining a target virtual object according to an embodiment of this application. Referring to FIG. 1, the implementation environment includes a terminal 110 and a server 120.

[0050] The terminal 110 and the server 120 may be directly or indirectly connected in a wired or wireless communication manner. This is not limited in this application.

[0051] The terminal 110 may be a smartphone, a tablet computer, a notebook computer, a desktop computer, a smart speaker, a smartwatch, or the like, but is not limited thereto. An application program supporting a virtual scene is installed and run on the terminal 110. The application program may be any one of a MOBA game, a massively multiplayer online role-playing game (MMORPG), a first-person shooter (FPS) game, a third-person shooter game, a virtual reality application program, a three-dimensional map program, a military simulation program, and a multiplayer gunfight survival game.

[0052] For example, the terminal 110 may be a terminal used by a user, and the application program running on the terminal 110 logs in to a user account. The user uses the terminal 110 to operate a controlled virtual object located in a virtual scene to perform activities. The activities

include, but are not limited to: at least one of casting skills, adjusting body postures, crawling, walking, running, riding, jumping, driving, picking, shooting, attacking, and throwing. For example, the controlled virtual object is a virtual character such as a simulated character role or a cartoon character role.

[0053] The server 120 may be an independent physical server, or may be a server cluster including a plurality of physical servers or a distributed system, or may be a cloud server providing basic cloud computing services, such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a content delivery network (CDN), big data, and an AI platform. The server 120 is configured to provide a backend service for an application program supporting a virtual scene.

[0054] Optionally, the server 120 may take on primary computing work, and the terminal 110 may take on secondary computing work; alternatively, the server 120 takes on secondary computing work, and the terminal 110 takes on primary computing work; alternatively, the server 120 or the terminal 110 may separately take on classification work; alternatively, collaborative computing is performed by using a distributed computing architecture between the server 120 and the terminal 110.

[0055] Optionally, the server 120 may be formed by an access server, a backend server, and a database server. The access server is configured to provide an access service for the terminal 110. The backend server is configured to provide a backend service of an application program of a virtual scene. There may be one or more backend servers. When there are a plurality of backend servers, at least two backend servers are configured to provide different services, and/or at least two backend servers are configured to provide the same service, for example, provide the same service in a load balancing manner. This is not limited in the embodiments of this application.

[0056] The terminal 110 may generally refer to one of a plurality of terminals. In this embodiment, the terminal 110 is merely used as an example for description. Application programs installed on the terminals may be the same, or the application programs installed on the two terminals are the same type of application programs on different operating system platforms. A person skilled in the art may learn that there may be more or fewer terminals. For example, there may be only one terminal, or there may be tens of or hundreds of terminals or more. The quantity and the device type of the terminals are not limited in the embodiments of this application.

[0057] In some embodiments, the controlled virtual object controlled by the terminal 110 and another virtual object controlled by another terminal are in the same virtual scene. In this case,

the controlled virtual object may perform adversarial interaction with the another virtual object in the virtual scene. The controlled virtual object and the another virtual object may be in a hostile relationship, for example, the controlled virtual object and the another virtual object may belong to different teams and camps. The virtual objects in the hostile relationship may battle against each other by casting skills, for example, the controlled virtual object casts an attack skill to the another virtual object. In some other embodiments, the controlled virtual object and the another virtual object may be alternatively in a teammate relationship, for example, a target virtual character and another virtual character may belong to the same team or the same organization, have a friend relationship, or have a temporary communication permission. In this case, the controlled virtual object may cast a healing skill to the another virtual object.

[0058] FIG. 2 is a flowchart of a method for determining a target virtual object according to an embodiment of this application. In this embodiment of this application, an example in which the method is applied to a terminal is used. The terminal may be the terminal 110 shown in FIG. 1. Referring to FIG. 2, the method for determining a target virtual object includes the following steps:

[0059] 201: The terminal determines a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user.

[0060] In this embodiment of this application, the controlled virtual object has at least one skill that may perform adversarial interaction with a first virtual object of a hostile object set (for example, a hostile camp). The user may trigger the skill by performing trigger operations such as clicking/tapping and dragging on the trigger option of the skill. When the target skill of the controlled virtual object is triggered, the terminal may obtain the trigger operation for the target skill. The terminal may use a position of the controlled virtual object as the center, and use a first virtual object found in a specific search range as an optional target of the target skill. Because different skills correspond to different search ranges, there are different quantities of first virtual objects in the target search range.

[0061] 202: The terminal obtains, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets.

[0062] In this embodiment of this application, the defeated condition indicates whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object. The target skill triggered by the controlled virtual object may

be an attack skill for reducing the virtual health points of the first virtual object. When the virtual health points of the first virtual object are reduced to zero, the first virtual object is defeated. If the target skill can reduce virtual health points of any first virtual object to zero, the first virtual object is a first virtual object that satisfies the defeated condition.

[0063] 203: The terminal obtains, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and takes the third virtual object as a target virtual object to be hit by the target skill.

[0064] In this embodiment of this application, the target hit condition indicates whether the third virtual object is used as a processing target of the target skill. The user may set a hit condition: for example, a virtual object has the lowest virtual health point percentage, a virtual object is closest to the controlled virtual object, or a virtual object has the lowest virtual health points. The terminal may select, according to the target hit condition set by the user, a third virtual object that satisfies the target hit condition as the target virtual object, that is, the target virtual object to be hit by the target skill. The target virtual object is defeated after being hit by the target skill.

[0065] In this embodiment of this application, the at least one second virtual object that satisfies the defeated condition, that is, can be defeated by the target skill, is obtained from at least two first virtual objects obtained within a search range corresponding to the target skill, and then the target hit condition set by the user is used for screening, so that the target virtual object determined by the terminal is consistent with a target virtual object intended by the user, thereby improving the efficiency of human-computer interaction.

[0066] FIG. 3 is a flowchart of a method for determining a target virtual object according to an embodiment of this application. In this embodiment of this application, an example in which the method is applied to a terminal is used. The terminal may be the terminal 110 shown in FIG. 1. Referring to FIG. 3, the method for determining a target virtual object includes the following steps:

[0067] 301: The terminal determines a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user.

[0068] In this embodiment of this application, during a game, the user may control the controlled virtual object to interact with other virtual objects in a virtual scene. The controlled virtual object has at least one skill that can be triggered, and the skill may be one of an attack skill, a defense skill, a healing skill, a carry skill, and an execution skill. The attack skill is generally shown as deducting part of the virtual health points of a hit first virtual object belonging to a hostile camp

after the first virtual object is hit. In addition, generally, the attack skill cannot select a second virtual object belonging to a friendly camp as a target to be hit. The user may trigger corresponding skills by triggering trigger options of different skills, and the terminal can detect the trigger operations when the skills are triggered.

[0069] A manner of triggering a skill is briefly described below.

[0070] In a MOBA game running on the terminal, a trigger option of a skill in the MOBA game is usually displayed in the form of a skill joystick. The skill joystick is a virtual button, including a first region and a second region. The first region is used for quickly triggering a skill corresponding to the skill joystick, and is usually referred to as quick casting. The second region is used for selecting, by the user through a drag operation, a target to be hit by the skill corresponding to the skill joystick, and is usually referred to as active aiming.

[0071] For example, FIG. 4 is a schematic diagram of a manner of triggering a skill according to an exemplary embodiment of this application. As shown in FIG. 4, 400 represents a skill joystick. The skill joystick includes a first region 401 and a second region 402. The first region 401 is a circle with a skill activation point being the center and a radius of r , and the second region 402 is an annular region outside the first region 401, r being a positive number. A user may click/tap the first region 401 to implement quick casting, and the user may press the skill joystick and drag the skill joystick to the second region 402 to implement active aiming.

[0072] When the user triggers a skill in different manners, the terminal determines a target search range corresponding to a target skill in different manners.

[0073] When the user triggers a skill in a manner of quick casting, the terminal may use a circular region with a current second scene position of the controlled virtual object being the center and a radius of N as the target search range corresponding to the skill, N being a positive number.

[0074] For example, FIG. 5 is a schematic diagram of a target search range according to an embodiment of this application. As shown in FIG. 5, a controlled virtual object 501, a first virtual object 502, and a first virtual object 503 are included in FIG. 5. A dashed-line region 504 is a circular region with a second scene position of the controlled virtual object 501 being the center and a radius of N , and the dashed-line region 504 is the target search range. The radius N may be a sum of a skill casting range parameter and a skill search range parameter.

[0075] When the user triggers a skill in a manner of active aiming, the terminal may determine the target search range corresponding to the skill according to a distance by which the user drags a trigger option and the current second scene position of the controlled virtual object.

[0076] Correspondingly, the step of determining, by the terminal, a target search range corresponding to a target skill may be as follows: The terminal may obtain a first screen position and a second screen position in response to a drag operation performed on a trigger option of the target skill, the first screen position being a start position, and the second screen position being an end position. The terminal may determine a first scene position corresponding to the target skill in a virtual scene according to the first screen position and the second screen position, the first scene position being a central position of the target search range. The terminal may determine the target search range corresponding to the target skill according to the first scene position, a second scene position of the controlled virtual object, and skill range information of the target skill, the skill range information including a skill casting range parameter and a skill search range parameter. A first scene position is determined according to a relative position of the trigger option dragged by the user, and the target search range corresponding to the target skill is determined with reference to the second scene position of the controlled virtual object and the skill range information, so that the target search range includes a first virtual object intended by the user, thereby improving the efficiency of human-computer interaction.

[0077] For example, FIG. 6 is a schematic diagram of another target search range according to an embodiment of this application. As shown in FIG. 6, a controlled virtual object 501, a first virtual object 502, a first virtual object 503, a dashed-line region 504 that is a circular region with a second scene position of the controlled virtual object 501 being the center and a radius of N , a first screen position 601, a second screen position 602, a first scene position 603, and a dashed-line region 604 that is a circular region with the first scene position 603 being the center and a radius of R are included in FIG. 6. The dashed-line region 604 is a target search range.

[0078] In an optional implementation, a distance between the first screen position and the second screen position may be mapped to a distance between a first scene position and a second scene position through a proportional relationship. Correspondingly, the step of determining, by a terminal, a first scene position corresponding to the target skill in a virtual scene according to the first screen position and the second screen position may be as follows: The terminal may determine the first position information according to the first screen position, the second screen position, and a third screen position corresponding to a screen center, the first position information being used for indicating a direction of the first scene position relative to the second scene position. Then, the terminal may determine second position information according to the first screen position, the second screen position, drag range information of the trigger option of the target skill, and the skill range information of the target skill, the second position information being used for indicating a distance between the first scene position and the second scene position. Finally, the terminal may

determine the first scene position corresponding to the target skill in the virtual scene according to the second scene position, the first position information, and the second position information.

[0079] Because a change from the first screen position to the second screen position is mapped to a change from the second scene position to the first scene position in the virtual scene, the first scene position is a position intended by a user, thereby improving the efficiency of human-computer interaction.

[0080] The terminal may express proportional relationships of distances between the positions by using the following Formula (1):

$$\frac{|DragPos - DownPos|}{MaxDragRadius} = \frac{|FocusPoint - HeroPos|}{SkillRange} \quad (1)$$

[0081] where *DownPos* represents a first screen position, *DragPos* represents a second screen position, *MaxDragRadius* represents a maximum drag range indicated by drag range information of a trigger option of a target skill, *FocusPoint* represents a first scene position, *HeroPos* represents a current second scene position of a controlled virtual object, and *SkillRange* represents a skill casting range indicated by skill range information of the target skill.

[0082] Based on the proportional relationships, the terminal may map a third screen position corresponding to a screen center (0, 0) to the virtual scene by using a center mapping method, to obtain a third scene position. Subsequently, a fourth scene position is obtained through mapping according to a sum of distance differences between the screen center and the first screen position and between the screen center and the second screen position. A direction from the third scene position to the fourth scene position is the same as a direction from the second scene position points to the first scene position. Therefore, the first scene position may be calculated by using the following Formula (2):

$$FocusPoint = HeroPos + \frac{|DragPos - DownPos|}{MaxDragRadius} \times SkillRange \quad (2)$$

$$\times Normalize(ScreenDrag2ScenePos - ScreenCenter2ScenePos)$$

[0083] where *FocusPoint* represents the first scene position, *HeroPos* represents a current second scene position of a controlled virtual object, *SkillRange* represents a skill casting range indicated by skill range information of the target skill, *Normalize*() represents

normalization, *ScreenCenter2ScenePos* represents the third scene position, and *ScreenDrag2ScenePos* represents the fourth scene position.

[0084] The terminal may invoke an application programming interface for searching for the first virtual object, to search for the first virtual object included in the target search range. The terminal may use a plurality of parameters such as a skill identifier, skill casting range information, and the skill range information of the target skill and the first scene position as input parameters of the application programming interface.

[0085] 302: The terminal determines, from the target search range according to a target filtering condition corresponding to the target skill, a first virtual object that satisfies the target filtering condition, the first virtual object and the controlled virtual object belonging to different object sets.

[0086] In this embodiment of this application, the terminal may store, in the form of a list, objects that can be used as targets of the target skill in the target search range, and obtain the target filtering condition corresponding to the target skill, to delete, from the list, an object that does not satisfy the target filtering condition, so that the remaining objects are the objects that satisfy the target filtering condition. The target filtering condition may be a condition set by a technician for the target skill, for example: an object type of a target is a virtual object, a target is a virtual object belonging to an enemy camp, a virtual object is in a visible state, and a virtual object is in a selectable state. This is not limited in this embodiment of this application. Different objects such as the first virtual object can be obtained through filtering by setting different filtering conditions.

[0087] For example, the target skill may act on a first virtual object belonging to the enemy camp, or may act on a building that belongs to the enemy camp, or may act on a neutral virtual object. The terminal may delete, from the list according to the target filtering condition, an object that does not belong to the first virtual object, a building that does not belong to the enemy camp, and an object that does not belong to the neutral virtual object. In another example, the target skill cannot act on virtual objects in an invisible state and an unselectable state, and the terminal may delete the virtual objects in an invisible state and an unselectable state from the list.

[0088] 303: The terminal determines, in response to that there is only one first virtual object in the target search range, the first virtual object as a target virtual object, and ends the procedure.

[0089] In this embodiment of this application, if there is only one first virtual object in the target search range, the terminal does not need to perform excessive determination on the first

virtual object. The first virtual object is the target virtual object intended by the user. The terminal may directly determine the first virtual object as the target virtual object.

[0090] 304: The terminal obtains, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects.

[0091] In this embodiment of this application, the target skill may cause damage to the first virtual object, to reduce virtual health points of the first virtual object. If the target skill can reduce the virtual health points of the first virtual object to zero, the target skill can defeat the first virtual object, that is, the first virtual object satisfies the defeated condition.

[0092] In an optional implementation, the target skill may cause various types of damage such as single damage, splash damage, physical damage, magic damage, and real damage. The damage types are not limited in this embodiment of this application.

[0093] Correspondingly, the step of obtaining, by the terminal, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects may be as follows: The terminal performs the following processing on any one of the at least two first virtual objects: obtaining at least one damage type corresponding to the target skill; accumulating a damage value corresponding to the at least one damage type, to obtain a total damage value; and determining, in response to that the total damage value is greater than current virtual health points of the first virtual object, the first virtual object as the second virtual object that satisfies the defeated condition.

[0094] Damage values of various types of damage caused by the target skill are added and it is then determined whether each first virtual object satisfies the defeated condition, the first virtual object that can be defeated by the target skill may be determined more accurately, thereby fulfilling the user and improving the efficiency of human-computer interaction.

[0095] For example, the target skill may cause damage of three types, namely, a damage type A, a damage type B, and a damage type C. The terminal may accumulate damage values a, b, and c caused by the three damage types, to obtain a total damage value, and it is determined through the total damage value whether the first virtual object can be defeated.

[0096] Correspondingly, for a procedure of determining, by the terminal, whether different first virtual objects satisfy the defeated condition, reference may be made to FIG. 7. FIG. 7 is a schematic flowchart of obtaining a first virtual object that satisfies a defeated condition according to an embodiment of this application. As shown in FIG. 7, the at least two first virtual objects

determined by the terminal within the target search range are stored in the form of a list, to obtain a target list.

[0097] Step 701: The terminal traverses the target list.

[0098] Step 702: The terminal initializes, for any first virtual object, a total damage value of the first virtual object, that is, sets HurtValue = 0.

[0099] Step 703: Traverse at least one damage type corresponding to a target skill.

[0100] Step 704: The terminal calculates a damage value curHurt corresponding to each damage type, and accumulates the damage values, that is, HurtValue = HurtValue + curHurt.

[0101] Step 705: Determine whether the total damage value is greater than current virtual health points of the first virtual object, if the total damage value is greater than the current virtual health points of the first virtual object, process a next first virtual object, and if the total damage value is not greater than the current virtual health points of the first virtual object, perform step 706.

[0102] Step 706: Delete the first virtual object from the target list if the total damage value is not greater than the current virtual health points of the first virtual object.

[0103] 305: Obtain, from the at least two first virtual objects in response to that none of the at least two first virtual objects satisfies the defeated condition, a third virtual object that satisfies a target hit condition set by the user, and take the third virtual object as a target virtual object to be hit by the target skill, and end the procedure.

[0104] In this embodiment of this application, before the game starts or during the game, the user may set different hit conditions, for example: a remaining virtual health point value takes priority, a virtual health point percentage takes priority, a distance to a controlled virtual object takes priority, and a distance to a first scene position takes priority. The terminal may determine, according to the target hit condition set by the user, the first virtual object that best satisfies the target hit condition from the at least two first virtual objects as the target virtual object.

[0105] If the target hit condition set by the user is that the remaining virtual health point value takes priority, that is, a first virtual object with the smallest remaining virtual health point value is preferentially selected as the target virtual object, the terminal may sort the at least two first virtual objects in ascending order according to current remaining virtual health point values, and then returns the first virtual object that ranks the first place.

[0106] In another example, if the target hit condition set by the user is that the virtual health point percentage takes priority, that is, a first virtual object with the smallest remaining virtual

health point percentage is preferentially selected as the target virtual object, the terminal may sort the at least two first virtual objects in ascending order according to current remaining virtual health point percentages, and then returns the first virtual object that ranks the first place.

[0107] Similarly, if the target hit condition is that the distance to the controlled virtual object takes priority, the first virtual object closest to the controlled virtual object is preferentially selected as the target virtual object. If the target hit condition is that the distance to the first scene position takes priority, the first virtual object closest to the first scene position is preferentially selected as the target virtual object. The target hit condition is not limited in this embodiment of this application.

[0108] For example, a first virtual object A has total virtual health points of 1000 and current virtual health points of 300. A first virtual object B has total virtual health points of 500 and current virtual health points of 200. When the target hit condition is that the remaining virtual health point value takes priority, the first virtual object B is the target virtual object. When the target hit condition is that the virtual health point percentage takes priority, the first virtual object A is the target virtual object.

[0109] 306: The terminal obtains, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and takes the third virtual object as a target virtual object to be hit by the target skill.

[0110] In this embodiment of this application, if the terminal obtains at least one second virtual object that can be defeated by the target skill, the terminal may further screen the at least one second virtual object according to the target hit condition set by the user. Reference may be made to step 305, and details are not described herein.

[0111] Step 301 to step 306 are an optional embodiment of the method for determining a target virtual object provided in the embodiments of this application, and correspondingly, there are other optional implementations. FIG. 8 is a schematic flowchart of another method for determining a target virtual object according to an embodiment of this application. The method is applied to a terminal, and the terminal may be the terminal 110 shown in FIG. 1.

[0112] A logic layer is provided in the terminal for logical operation of data and transmitting the data obtained after the operation to a presentation layer for execution. The presentation layer is further provided in the terminal for managing an input of the user through an operation interface, transmitting a data packet to a server, and executing the data transmitted by the

logic layer. The server is disposed on a server side for receiving the data transmitted in the presentation layer and synchronizing the data to a logic layer of a client.

[0113] As shown in FIG. 8, the method includes the following steps:

[0114] Step 801: The terminal obtains information of each virtual object in the logic layer.

[0115] Step 802: Search for whether a valid target is included in a target search range of a target skill, that is, a virtual object that can be hit by the target skill, and if no valid target is included, perform step 803, or if a valid target is included, perform step 804.

[0116] Step 803: Continuously search for a valid target.

[0117] Step 804: Determine whether there is a valid target that can be defeated by the target skill, and if not, perform step 805, or if yes, perform step 806.

[0118] Step 805: Perform a default target search procedure, that is, select a target virtual object according to a target hit condition.

[0119] Step 806: Sort, according to the target hit condition set by a user, valid targets that can be defeated.

[0120] Step 807. Select a valid target with the highest priority as the target virtual object.

[0121] In this embodiment of this application, at least one second virtual object that can be defeated by the target skill is obtained from at least two first virtual objects obtained within a search range corresponding to the target skill, and then the target hit condition set by the user is used for screening, so that the target virtual object determined by the terminal is consistent with a target virtual object intended by the user, thereby improving the efficiency of human-computer interaction.

[0122] FIG. 3 shows an implementation of a method for determining a target virtual object according to an embodiment of this application. Correspondingly, in this embodiment of this application, a target virtual object may be determined through complex search logic, and the search logic may be referred to as a search tree.

[0123] FIG. 9 is a structural diagram of a class diagram of a search tree according to an embodiment of this application. As shown in FIG. 9, all nodes in the search tree are inherited from a base selector. The base selector includes three core derived subclasses: a link selector, a branch selector, and a priority selector.

[0124] The base selector mainly includes two methods: a configuration function Configure () and a selection function Select (). The configuration function is used for initializing data of

selectors through table data configured by a technician. For example, the branch selector needs to be configured with a plurality of branches, and data configured for the configuration function is IDs of several branch selectors. In another example, a subclass wave-shape filter of the link selector needs to be configured with shape fields, such as a circle, a sector, a circle radius, a sector angle, and other parameters. An input parameter of the selection function is List<BattleActor>, which is a list storing virtual objects, and a return parameter is BattleActor, that is, a virtual object.

[0125] The following introduces the three core derived subclasses of the base selector respectively.

[0126] Link selector: the core of the link selector is a next parameter, which is used for indicating a next required filter, to form a chain-like structure. The link selector includes a plurality of subclasses. The plurality of subclasses are basically filters and mainly configured to delete, from the parameters in the selection function, virtual objects that do not satisfy conditions, and transfer the List<BattleActor> with the virtual objects deleted to a next selector. In this way, the virtual objects are screened.

[0127] For example, the wave-shape filter configures required graphics and parameters in the configuration function. The selection function determines whether the virtual objects in the parameter List<BattleActor> are within a shape range one by one, and deletes the virtual objects that are not in the range from the list. Other filters perform similar operations.

[0128] For example, a buffer-type filter deletes virtual objects that include a type of additional state such as an invisible state or an unselectable state.

[0129] For example, an identity filter (ID filter) deletes a virtual object that includes a skill identifier, and is used for processing to prevent a skill from hitting an enemy a second time.

[0130] For example, a can-kill filter is configured to determine whether a skill can defeat the virtual object. In addition to the foregoing filters, there are many other filters, and details are not described herein again.

[0131] Branch selector: A main function of the branch selector is to process a plurality of condition priorities. A plurality of selector IDs are configured in a configuration table. In the configuration function, a member variable selector set is initialized according to the configured selector IDs. In the selection function, the virtual objects in the parameter List<BattleActor> need to be temporarily stored, and the selection function is invoked with the temporarily stored List<BattleActor> as a parameter by using the base selectors in the selector set one by one, to determine whether BattleActor is returned. If BattleActor is returned, it indicates that one virtual

object satisfies a target condition, for example, one virtual object is within a target search range or can be hit by a target skill, and the subsequent selector set does not need to be traversed. If BattleActor is not returned, a base selector in a next selector set needs to be traversed.

[0132] Priority selector: configured to sort the filtered List<BattleActor> to select an appropriate virtual object. A technician can configure hit conditions in a table, for example: health points take priority, a distance takes priority, a health point percentage takes priority, and in a distance to a focus point takes priority, and a user selects the hit conditions. In the selection function, the parameter List<BattleActor> is sorted according to the configured hit conditions, and the first one in the list is returned. If the list is empty, NULL is returned.

[0133] Complex search logic can be implemented by combining a plurality of selectors, and a target virtual object that fulfills the intention of the user is obtained. For example, FIG. 10 is a schematic diagram of a game scene according to an embodiment of this application. As shown in FIG. 10, a controlled virtual object 1001, a first virtual object 1002, a first virtual object 1003, and a first virtual object 1004 are included. The first virtual object 1002 has the lowest current virtual health points but has relatively high magic resistance, that is, can reduce damage caused by a magic attack. The first virtual object 1003 and the first virtual object 1004 have relatively high current virtual health points but relatively low magic resistance. When a target skill triggered by the controlled virtual object 1001 is a magic attack skill, the first virtual object 1003 and the first virtual object 1004 can be defeated, but the first virtual object 1002 cannot be defeated.

[0134] The user intends to defeat one of the first virtual objects through the target skill, and then logic of a search tree may be shown in FIG. 11. FIG. 11 is a schematic logic diagram of a search tree according to an embodiment of this application. As shown in FIG. 11, three first virtual objects, namely, a first virtual object 1002, a first virtual object 1003, and a first virtual object 1004, are first determined by using a branch selector. Subsequently, the three first virtual objects are inputted to a can-kill filter to filter out the first virtual object that cannot be defeated by the target skill. Because the first virtual object 1002 cannot be defeated by the target skill, through the can-kill filter, the first virtual object 1003 and the first virtual object 1004 are obtained. Filtering is then performed through PrioritySelector+PriorityType: HPLow (the lowest virtual health points), that is, a target hit condition set by the user: virtual health points take priority. Because current virtual health points of the first virtual object 1003 are less than current virtual health points of the first virtual object 1004, the first virtual object 1003 is returned as the target virtual object.

[0135] Simplified search logic is set for different virtual objects and different skills through the search tree, so that the user can quickly and accurately select an intended virtual object through

a relatively low operating cost, thereby improving the efficiency of human-computer interaction. Especially for some skills with special mechanisms, for example, skills that can be reused after a virtual object belonging to a hostile camp is defeated, in the method provided in the embodiments of this application, a success rate of reusing the skills can be greatly increased and the efficiency of human-computer interaction is improved, thereby bringing better user experience to users and improving the playability of a game.

[0136] FIG. 12 is a schematic structural diagram of an apparatus for determining a target virtual object according to an embodiment of this application. Referring to FIG. 12, the apparatus includes a range determining module 1201, a first object obtaining module 1202, and a second object obtaining module 1203.

[0137] The range determining module 1201 is configured to determine a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user.

[0138] The first object obtaining module 1202 is configured to obtain, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets, the defeated condition indicating whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object.

[0139] The second object obtaining module 1203 is configured to obtain, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and take the third virtual object as a target virtual object to be hit by the target skill.

[0140] In an optional implementation, the first object obtaining module 1202 is configured to perform the following processing on any one of the at least two first virtual objects: obtaining at least one damage type corresponding to the target skill; accumulating a damage value corresponding to the at least one damage type, to obtain a total damage value; and determining, in response to that the total damage value is greater than current virtual health points of the first virtual object, the first virtual object as the second virtual object that satisfies the defeated condition.

[0141] In an optional implementation, the range determining module 1201 includes:

[0142] a position obtaining unit 12011, configured to obtain a first screen position and a second screen position in response to a drag operation performed on a trigger option of the target

skill, the first screen position being a start position, and the second screen position being an end position;

[0143] a position determining unit 12012, configured to determine a first scene position corresponding to the target skill in a virtual scene according to the first screen position and the second screen position, the first scene position being a central position of the target search range; and

[0144] a range determining unit 12013, configured to determine the target search range corresponding to the target skill according to the first scene position, a second scene position of the controlled virtual object, and skill range information of the target skill, the skill range information including a skill casting range parameter and a skill search range parameter.

[0145] In an optional implementation, the position determining unit 12012 is configured to: determine first position information according to the first screen position, the second screen position, and a third screen position corresponding to a screen center, the first position information being used for indicating a direction of the first scene position relative to the second scene position; determine second position information according to the first screen position, the second screen position, drag range information of the trigger option of the target skill, and the skill range information of the target skill, the second position information being used for indicating a distance between the first scene position and the second scene position; and determine the first scene position according to the second scene position, the first position information, and the second position information.

[0146] In an optional implementation, the apparatus further includes:

[0147] a condition obtaining module 1204, configured to obtain a target filtering condition corresponding to the target skill; and

[0148] a third object obtaining module 1205, configured to determine, from the target search range, a first virtual object that satisfies the target filtering condition.

[0149] In an optional implementation, the second object obtaining module 1203 is further configured to obtain, from the at least two first virtual objects in response to that none of the at least two first virtual objects satisfies the defeated condition, the third virtual object that satisfies the target hit condition set by the user, and take the third virtual object as the target virtual object to be hit by the target skill.

[0150] In an optional implementation, the second object obtaining module 1203 is further configured to determine, in response to that there is only one first virtual object in the target search range corresponding to the target skill, the first virtual object as the target virtual object.

[0151] In this embodiment of this application, the at least one second virtual object that satisfies the defeated condition, that is, can be defeated by the target skill, is obtained from at least two first virtual objects obtained within a search range corresponding to the target skill, and then the target hit condition set by the user is used for screening, so that the target virtual object determined by the terminal is consistent with a target virtual object intended by the user, thereby improving the efficiency of human-computer interaction.

[0152] FIG. 13 is a structural block diagram of a terminal 1300 according to an embodiment of this application. The terminal 1300 may be a smartphone, a tablet computer, a Moving Picture Experts Group Audio Layer III (MP3) player, a Moving Picture Experts Group Audio Layer IV (MP4) player, a notebook computer, or a desktop computer. The terminal 1300 may also be referred to other names such as user equipment, a portable terminal, a laptop terminal, or a desktop terminal.

[0153] Generally, the terminal 1300 includes a processor 1301 and a memory 1302.

[0154] The processor 1301 may include one or more processing cores, for example, a 4-core processor or an 8-core processor. The processor 1301 may be implemented in at least one hardware form of a digital signal processor (DSP), a field-programmable gate array (FPGA), and a programmable logic array (PLA). The processor 1301 may alternatively include a main processor and a coprocessor. The main processor is a processor configured to process data in an awake state, also referred to as a central processing unit (CPU), and the coprocessor is a low-power processor configured to process data in a standby state. In some embodiments, the processor 1301 may be integrated with a graphics processing unit (GPU). The GPU is configured to be responsible for rendering and drawing content that a display screen needs to display. In some embodiments, the processor 1301 may further include an AI processor. The AI processor is configured to process a computing operation related to machine learning.

[0155] The memory 1302 may include one or more computer-readable storage medium. The computer-readable storage medium may be non-transient. The memory 1302 may further include a high-speed random access memory and a non-volatile memory, such as one or more magnetic disk storage devices or a flash storage device. In some embodiments, the non-transient computer-readable storage medium in the memory 1302 is configured to store at least one instruction. The at least one instruction is executed by the processor 1301 to perform the method for determining a target virtual object provided in the method embodiment in this application.

[0156] In some embodiments, the terminal 1300 may optionally include a peripheral interface 1303 and at least one peripheral. The processor 1301, the memory 1302, and the peripheral interface 1303 may be connected by using a bus or a signal cable. Each peripheral may be connected to the peripheral interface 1303 by using a bus, a signal cable, or a circuit board. Specifically, the peripheral includes at least one of a radio frequency circuit 1304, a display screen 1305, a camera assembly 1306, an audio circuit 1307, a positioning component 1308, and a power supply 1309.

[0157] In some embodiments, the terminal 1300 may also include one or more sensors 1310. The one or more sensors 1310 include, but are not limited to, an acceleration sensor 1311, a gyroscope sensor 1312, a pressure sensor 1313, a fingerprint sensor 1314, an optical sensor 1315, and a proximity sensor 1316.

[0158] A person skilled in the art may understand that the structure shown in FIG. 13 does not constitute a limitation on the terminal 1300, and the terminal may include more components or fewer components than those shown in the figure, or some components may be combined, or a different component deployment may be used.

[0159] An embodiment of this application further provides a computer-readable storage medium. The computer-readable storage medium is applied to a terminal and includes a memory of at least one piece of program code, the at least one piece of program code being executed by a processor in the terminal to complete the method for determining a target virtual object in the foregoing 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, or the like.

[0160] An embodiment of this application further provides an application program product, storing one or more instructions, the one or more instructions being executed by a processor of a computer device to implement the method for determining a target virtual object.

[0161] 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 hardware, or may be implemented a program instructing related hardware. The program may be stored in a computer-readable storage medium. The storage medium may be: a ROM, a magnetic disk, or an optical disc.

[0162] The foregoing descriptions are merely optional embodiments of this application, but are not intended to limit this application. Any modification, equivalent replacement, or

improvement made within the spirit and principle of this application shall fall within the protection scope of this application.

CLAIMS

What is claimed is:

1. A method for determining a target virtual object, performed by a terminal, the method comprising:

determining a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user;

obtaining, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets, the defeated condition indicating whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object; and

obtaining, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and taking the third virtual object as a target virtual object to be hit by the target skill.

2. The method according to claim 1, wherein obtaining the at least one second virtual object that satisfies the defeated condition from the at least two first virtual objects comprises:

performing the following processing on any one of the at least two first virtual objects:

obtaining at least one damage type corresponding to the target skill;

accumulating a damage value corresponding to the at least one damage type, to obtain a total damage value; and

determining, in response to that the total damage value is greater than current virtual health points of the first virtual object, the first virtual object as the second virtual object that satisfies the defeated condition.

3. The method according to claim 1, wherein determining the target search range corresponding to the target skill comprises:

obtaining a first screen position and a second screen position in response to a drag operation performed on a trigger option of the target skill, the first screen position being a start position, and the second screen position being an end position;

determining a first scene position corresponding to the target skill in a virtual scene according

to the first screen position and the second screen position, the first scene position being a central position of the target search range; and

determining the target search range according to the first scene position, a second scene position of the controlled virtual object, and skill range information of the target skill, the skill range information comprising a skill casting range parameter and a skill search range parameter.

4. The method according to claim 3, wherein determining the first scene position corresponding to the target skill in the virtual scene according to the first screen position and the second screen position comprises:

determining first position information according to the first screen position, the second screen position, and a third screen position corresponding to a screen center, the first position information being used for indicating a direction of the first scene position relative to the second scene position;

determining second position information according to the first screen position, the second screen position, drag range information of the trigger option of the target skill, and the skill range information of the target skill, the second position information being used for indicating a distance between the first scene position and the second scene position; and

determining the first scene position according to the second scene position, the first position information, and the second position information.

5. The method according to claim 3, further comprising: after determining the target search range corresponding to the target skill,

obtaining a target filtering condition corresponding to the target skill; and

determining, from the target search range, a first virtual object that satisfies the target filtering condition.

6. The method according to claim 1, further comprising: before obtaining the at least one second virtual object that satisfies the defeated condition from the at least two first virtual objects,

obtaining, from the at least two first virtual objects in response to that none of the at least two first virtual objects satisfies the defeated condition, the third virtual object that satisfies the target hit condition set by the user, and taking the third virtual object as the target virtual object to be hit by the target skill.

7. The method according to claim 1, further comprising: after determining the target search range corresponding to the target skill,

determining, in response to that there is only one first virtual object in the target search range

corresponding to the target skill, the first virtual object as the target virtual object.

8. An apparatus for determining a target virtual object, comprising:

a range determining module, configured to determine a target search range corresponding to a target skill of a controlled virtual object in response to a trigger operation for the target skill, the controlled virtual object being a virtual object controlled by a user;

a first object obtaining module, configured to obtain, in response to that there are at least two first virtual objects in the target search range, at least one second virtual object that satisfies a defeated condition from the at least two first virtual objects, the first virtual object and the controlled virtual object belonging to different object sets, the defeated condition indicating whether virtual health points of the first virtual object reach a preset threshold after the target skill is performed on the first virtual object; and

a second object obtaining module, configured to obtain, from the at least one second virtual object, a third virtual object that satisfies a target hit condition set by the user, and take the third virtual object as a target virtual object to be hit by the target skill.

9. The apparatus according to claim 8, wherein the first object obtaining module is configured to perform the following processing on any one of the at least two first virtual objects: obtaining at least one damage type corresponding to the target skill; accumulating a damage value corresponding to the at least one damage type, to obtain a total damage value; and determining, in response to that the total damage value is greater than current virtual health points of the first virtual object, the first virtual object as the second virtual object that satisfies the defeated condition.

10. The apparatus according to claim 8, wherein the range determining module comprises:

a position obtaining unit, configured to obtain a first screen position and a second screen position in response to a drag operation performed on a trigger option of the target skill, the first screen position being a start position, and the second screen position being an end position;

a position determining unit, configured to determine a first scene position corresponding to the target skill in a virtual scene according to the first screen position and the second screen position, the first scene position being a central position of the target search range; and

a range determining unit, configured to determine the target search range according to the first scene position, a second scene position of the controlled virtual object, and skill range information of the target skill, the skill range information comprising a skill casting range parameter and a skill search range parameter.

11. The apparatus according to claim 10, wherein the position determining unit is configured to: determine first position information according to the first screen position, the second screen position, and a third screen position corresponding to a screen center, the first position information being used for indicating a direction of the first scene position relative to the second scene position; determine second position information according to the first screen position, the second screen position, drag range information of the trigger option of the target skill, and the skill range information of the target skill, the second position information being used for indicating a distance between the first scene position and the second scene position; and determine the first scene position according to the second scene position, the first position information, and the second position information.

12. The apparatus according to claim 10, further comprising:

a condition obtaining module, configured to obtain a target filtering condition corresponding to the target skill; and

a third object obtaining module, configured to determine, from the target search range, a first virtual object that satisfies the target filtering condition.

13. The apparatus according to claim 8, wherein the second object obtaining module is further configured to obtain, from the at least two first virtual objects in response to that none of the at least two first virtual objects satisfies the defeated condition, the third virtual object that satisfies the target hit condition set by the user, and take the third virtual object as the target virtual object to be hit by the target skill.

14. A terminal, comprising one or more processors and one or more memories, the one or more memories storing at least one piece of program code, the at least one piece of program code being loaded and executed by the one or more processors to implement the operations performed in the method for determining a target virtual object according to any one of claims 1 to 7.

15. A storage medium, storing at least one piece of program code, the at least one piece of program code being loaded and executed by a processor to implement the operations performed in the method for determining a target virtual object according to any one of claims 1 to 7.