

Method for Selecting Virtual Objects, Apparatus, Terminal and Storage Medium

This application claims the benefit of Chinese Patent Application, No. 202010398569.3, entitled "Method for Selecting Virtual Objects, Apparatus, Terminal and Storage medium", filed on May 12, 2020, the entire contents of which are hereby
5 incorporated by reference.

Technical Field

The application relates to computer and Internet technologies, and more
10 particularly, to a method for selecting virtual objects, an apparatus, a terminal and a storage medium.

Background

At present, there are more and more types of skills of virtual objects in game applications.

15 In related technologies, in a game, a user may control a first virtual object to use skills to attack a second virtual object, in which the second virtual object and the first virtual object are in different camps. When the first virtual object releases a skill, a client may obtain an attack scope of the skill, and takes the second virtual object within the attack scope as an attack target of the first virtual object.

20 However, in the foregoing related technologies, the selected attack target may be not displayed on a terminal display, which is inconsistent with the user's situation, resulting in a low accuracy in the selection of attack targets.

Summary

Embodiments of the application provide a method for selecting virtual objects on a
25 user interface of a terminal, an apparatus, the terminal and a storage medium, which may

ensure that the action target of the first operation is within the user's visual scope, and the accuracy for selecting the action target of the first operation is improved. The technical solutions are as follows.

In one aspect, the embodiments of the application provide a method for selecting a virtual object on a user interface of a terminal, performed by the terminal, including:
5 displaying a user interface, wherein the user interface comprises a display screen corresponding to a virtual environment, and further comprises a first virtual object located in the virtual environment; obtaining a predefined execution area of a first operation of the first virtual object within the virtual environment; obtaining a visible boundary scope
10 of the virtual environment, wherein virtual objects that reside in the visible boundary scope are visible on the user interface; determining, based on the predefined execution area and the visible boundary scope, an effective execution area of the first operation within the virtual environment; and, determining, based on the effective execution area, a target virtual object.

In another aspect, the embodiments of the application provide an apparatus for selecting a virtual object on a user interface of a terminal, including: an interface display module, configured to display a user interface, wherein the user interface comprises a display screen corresponding to a virtual environment, and further comprises a first virtual object located in the virtual environment; a scope obtaining module, configured to
20 obtain a predefined execution area of a first operation of the first virtual object within the virtual environment; a boundary obtaining module, configured to obtain a visible boundary scope of the virtual environment, wherein virtual objects that reside in the visible boundary scope are visible on the user interface; a scope determining module, configured to determine, based on the predefined execution area and the visible boundary scope, an effective execution area of the first operation within the virtual environment;
25 and, an object determining module, configured to determine, based on the effective execution area, a target virtual object.

In still another aspect, the embodiments of the application provide a terminal, including a processor and a memory, wherein the memory stores at least one instruction,

at least one program, a code set or an instruction set, when being loaded and executed by the processor, causing the processor to perform the above-mentioned method for selecting a virtual object.

In still another aspect, the embodiments of the application provide a computer
5 readable storage medium, wherein the computer readable storage medium stores at least one instruction, at least one program, a code set or an instruction set, when being loaded and processed by a processor, causing the processor to perform the above-mentioned method for selecting a virtual object.

In still another aspect, a computer program product is provided. When the computer
10 program product runs on a terminal, cause the terminal to perform foregoing method for selecting a virtual object.

Brief Descriptions of the Drawings

In order to more clearly describe the technical solutions in the embodiments of the application, attached figures needing to be used when describing the embodiments will be
15 briefly discussed in the following. It is obvious that, the following described attached figures are only some embodiments of the application. For persons having ordinary skill in the art, other attached figures may be obtained based on these attached figures without creative work.

FIG.1 is a schematic diagram illustrating a running environment of an application, in
20 accordance with an embodiment of the application.

FIG.2 is a schematic diagram illustrating structure of a terminal, in accordance with an embodiment of the application.

FIG.3 is a flowchart illustrating a method for selecting a virtual object on a user interface of a terminal, in accordance with an embodiment of the application.

25 FIG.4 is an exemplary schematic diagram illustrating how to obtain a predefined execution area.

FIG.5 is an exemplary schematic diagram illustrating how to display a visible boundary scope.

FIG.6 is an exemplary schematic diagram illustrating how to obtain an effective execution area.

FIG.7 is an exemplary schematic diagram illustrating how to obtain a visible boundary scope.

5 FIG.8 is a flowchart illustrating a method for selecting a virtual object on a user interface of a terminal, in accordance with another embodiment of the application.

FIG.9 is an exemplary schematic diagram illustrating a method for selecting a virtual object on a user interface of a terminal.

10 FIG.10 is a block diagram of an apparatus for selecting a virtual object on a user interface of a terminal, in accordance with an embodiment of the application.

FIG.11 is a block diagram of an apparatus for selecting a virtual object on a user interface of a terminal, in accordance with another embodiment of the application.

FIG.12 is a block diagram illustrating structure of a terminal, in accordance with an embodiment of the application.

15 Detailed Descriptions

To make objectives, technical solutions and advantages of the application more clear, detailed descriptions will be provided for implementations of the application in the following, accompanying with attached figures.

20 Please refer to FIG.1, FIG.1 is a schematic diagram illustrating a running environment of an application, in accordance with an embodiment of the application. The App running environment may include a terminal 10 and a server 20.

25 The terminal 10 may be electronic equipment (UE), such as a mobile phone, a tablet computer, a game host, an E-book reader, Multimedia playback equipment, a Wearable Device, a Personal Computer (PC). A client of an application may be installed to the terminal 10.

In embodiments of the application, foregoing application may be any application that can provide a virtual environment, such that a virtual object selected and operated by a user performs activities within the virtual environment. Typically, the application is a

game application, such as, Multiplayer Online Battle Arena (MOBA) game, Battle Royale (BR) game, Third-Personal Shooting Game (TPS), First-Person Shooting Game (FPS) and Multiplayer gun battle survival game. Of course, in addition to game applications, other types of applications may also display a virtual object for users, and provide a
5 corresponding function for the virtual object, e.g., Virtual Reality (VR) applications, Augmented Reality (AR) applications, three-dimensional map programs, military simulation programs, social applications, interactive entertainment applications, and so on, which are not limited by embodiments of the application. In addition, for a different application, the form and corresponding function of a virtual object provided will be
10 different, which may be pre-configured according to actual needs, and are not limited by embodiments of the application. A client of foregoing application runs on the terminal 10. In some embodiments, foregoing application is developed based on a three-dimensional virtual environment engine, e.g., such virtual environment engine is Unity engine. The virtual environment engine may build a third-dimensional virtual environment, a virtual
15 object and virtual props, and so on, which bring users a more immersive gaming experience.

The above-mentioned virtual environment is a scene, which is displayed (or provided) by a client of an application (e.g., a game application), when the client runs on the terminal. Such virtual environment refers to a scene created for virtual objects to
20 perform activities (such as game competition), such as virtual houses, virtual islands, virtual maps, and virtual buildings, and so on. Such virtual environment may be a simulation environment of the real world, a semi-simulation and semi-fictional environment, or a purely fictional environment. The virtual environment may be a two-dimensional virtual environment, or a 2.5-dimensional virtual environment, or a
25 three-dimensional virtual environment, which are not limited by embodiments of the application.

The above-mentioned virtual object may be a virtual character controlled by a user account in the application, or may be a virtual character controlled by a computer program in the application. In an example, where the application is a game application,

the virtual object may be a game character controlled by a user account in the game application, or may be a game monster controlled by a computer program in the game application. A virtual object may be a character form, an animal, a cartoon, or other form, which is not limited in the embodiments of the application. The virtual object may be displayed in a three-dimensional form or a two-dimensional form, which is not limited in the embodiments of the application. When the virtual environment is a three-dimensional environment, the virtual object is a three-dimensional model created based on animation skeletal technology. In the three-dimensional environment, each virtual object has its own shape and volume, and occupies a part of the space in the three-dimensional virtual environment. In the embodiments of the application, the terminal 10 may receive an operation from a virtual object, determine an effective execution area, based on a predefined execution area of the operation and a visible boundary scope of the virtual environment, and then, determine a target virtual object corresponding to the above-mentioned operation in the effective execution area.

In a possible implementation, the above-mentioned virtual objects include virtual objects in the same camp and virtual objects in different camps. That is, in the application, virtual objects are divided into different camps. Illustratively, taking the MOBA game as an example, ten users play a matching game and are divided into two teams, i.e., the red team and the blue team. That is, every 5 people are in a group. At this time, virtual objects of the red team are in the same camp and are teammates, and virtual objects of the blue team are in the same camp and are teammates. However, virtual objects of the red team and virtual objects of the blue team are in different camps, and are hostile to each other.

The server 20 is configured to provide background services for a client of an application in the terminal 10. For example, the server 20 may be a background server of foregoing application. The server 20 may be a server, or a server cluster composed of multiple servers, or a cloud computing service center. The server 20 may simultaneously provide background services for applications of various terminals 10.

The terminal 10 and the server 20 may communicate with each other through a network 30.

Please refer to FIG.2, FIG.2 is a schematic diagram illustrating structure of a terminal, in accordance with an embodiment of the application. The terminal 10 may include a main board 110, an external output/input device 120, a memory 130, an external interface 140, a touch control system 150 and a power supply 160.

5 The main board 110 integrates processing elements such as a processor and a controller.

For the terminal, the external output/input device 120 may include a display component (such as a display), a sound playback component (such as a speaker), a sound collection component (such as a microphone), all kinds of buttons, and so on. For a PC
10 terminal, the external output/input device 120 may include a display component (such as a display), a sound playback component (such as a speaker), a sound collection component (such as a microphone), all kinds of buttons (such as mouse and keyboard), and so on.

The memory 130 stores program codes and data.

15 The external interface 140 may include a headphone interface, a charging interface, a data interface, and so on.

The touch control system 150 may be integrated into the display component, or button of the external output/input device 120. The touch control system 150 is configured to detect a touch control operation performed by a user on the display
20 component or button.

The power supply 160 is configured to provide power for other components in the terminal 10.

In the embodiments of the application, the processor in the main board 110 may generate a user interface (e.g., game interface), by executing or calling program codes or
25 data in the memory, and display the generated user interface (e.g., game interface) through the external output/input device 120. During the process of displaying the user interface (e.g., game interface), a touch operation executed when a user interacts with a user interface (e.g., game interface) may be detected through the touch control system 150, and a response to the touch control operation may be made through the touch control

system 150.

Please refer to FIG.3, FIG.3 is a flowchart illustrating a method for selecting a virtual object on a user interface of a terminal, in accordance with an embodiment of the application. The method may be applied to the terminal, e.g., an execution entity of each
5 block may be the terminal 10 (referred to as “client” hereinafter) in the APP running environment shown in FIG.1. The method may include the following blocks (301~305).

In block 301, display a user interface.

The user interface may be a picture of an application displayed for a user by a client. The application may be a computer program needing to be downloaded and installed, and
10 may be a click-to-run computer program. Foregoing application may be any application that may provide a virtual environment, e.g., a game application program. The above-mentioned user interface includes a display screen of the virtual environment. The display screen may include virtual items, e.g., a virtual building, a virtual environment, a virtual map, and so on. The user may control a first virtual object to interact with various
15 virtual items within the virtual environment.

In the embodiments of the application, when receiving an instruction for triggering application launch from a user, the client may also control the running of the application, and display the user interface corresponding to the application. The above-mentioned user interface includes a display screen corresponding to the virtual environment, and further
20 includes a first virtual object located in the virtual environment. The first virtual object may be a virtual object controlled by the user.

In block 302, obtain a predefined execution area of a first operation of the first virtual object within the virtual environment.

The first virtual object is a virtual character controlled by foregoing user within the
25 virtual environment. The first virtual object may be in the form of a character, an animal, a cartoon, or other forms, which are not limited by the embodiments of the application. In the embodiments of the application, the first virtual object may be displayed in three-dimensional or two-dimensional form.

The first operation refers to any interactive operation between the first virtual object

and foregoing virtual environment, e.g., the first operation is a skill release operation of the first virtual object, or an ordinary attack operation of the first virtual object. The action target of the interactive operation may be a virtual item within the virtual environment, or other virtual objects of the virtual environment, which are not limited by the embodiments of the application.

In a possible implementation, foregoing first operation is triggered by the first virtual object, which is controlled by the user. The user may start the trigger instruction of the first operation, by clicking a corresponding icon or pressing a corresponding key. Furthermore, after receiving the trigger instruction, the client may control the first virtual object to perform foregoing first operation.

In another possible implementation, foregoing first operation is triggered by the first virtual object, which is controlled by the computer program. After running foregoing application, the client may control the first virtual object to perform foregoing first operation, according to a preset criteria. The preset criteria may be set flexibly. For example, foregoing preset criteria includes online duration of the first virtual object. The client may detect the online duration of the first virtual object. When the online duration of the first virtual object meets a requirement, the client may control the first virtual object to perform foregoing first operation. For another example, foregoing preset criteria includes attribute parameters of the first virtual object, the client may detect the attribute parameters of the first virtual object. When the one or more attribute parameters of the first virtual object meets a requirement, the client controls the first virtual object to perform foregoing first operation. The above-mentioned attribute parameters may include a life value, a defense value, or an attack speed, and so on, which are not limited by the embodiments of the application. Alternatively, foregoing preset criteria includes operation parameters of the first virtual object, and the client may detect the operation parameters of the first virtual object. When the operation parameters of the first virtual object meet a requirement, the client controls the first virtual object to perform foregoing first operation. Foregoing operation parameters may be number of monsters defeated, number of skill released, number of items used, and so on, which are not limited by the embodiments of

the application. It should be noted that, foregoing introduction to the preset criteria is only exemplary and explanatory. In practical applications, foregoing preset criteria may be flexibly set according to actual situations.

The predefined execution area refers to an action scope of foregoing first operation.
5 A different first operation may correspond to a different predefined execution area. In the embodiments of the application, after displaying foregoing user interface, the client may obtain the predefined execution area of the first operation of the first virtual object within the virtual environment. The shape of the predefined execution area may be circle, rectangle, or triangle, and so on, which are not limited by the embodiments of the
10 application.

In a possible implementation, foregoing predefined execution area is a preset scope. The preset scope is set by a designer, which is not changed during the running of the application. In another possible implementation, foregoing predefined execution area is a requirement scope. The requirement scope changes according to different requirements
15 during the running of the application, e.g., the predefined execution area of the first operation may be proportional or inversely proportional to the number of executions of the first operation, etc. In actual applications, foregoing requirement may be set flexibly according to actual situations, which are not limited by the embodiments of the application.

20 In the embodiments of the application, the client may obtain foregoing predefined execution area, according to location information of the first virtual object. Foregoing block 302 may include several blocks as follows.

1. Obtain location information of the first virtual object within the virtual environment.
- 25 2. Determine the predefined execution area of the first operation within the virtual environment, based on the location information and a predefined execution distance of the first operation.

The location information is configured to indicate the location of the virtual object within the virtual environment. The location information is represented in the form of

coordinates.

The predefined execution distance refers to the maximum action distance of foregoing first operation. Within the virtual environment, a virtual item or virtual object, distance to the first virtual object is less than the maximum action distance, may be taken
5 as the action target of the first operation. In the embodiments of the application, the client may obtain the location information of foregoing first virtual object, and determine the predefined execution area of the first operation, based on the location information and the predefined execution distance of the first operation. The client may determine boundary points of the predefined execution area, based on the location information and the
10 predefined execution distance of the first operation, and then determine the predefined execution area. Alternatively, the client may also determine each area point within the predefined execution area, based on the location information and the predefined execution distance of the first operation, and then determine the predefined execution area, which are not limited by the embodiments of the application. The same first operation may
15 possess one or more predefined execution distances.

In a possible implementation, the first operation has one predefined execution distance. At this time, the predefined execution area is a circle scope centered on the first virtual object. The distance between a boundary point of the predefined execution area and the first virtual object is the same. When obtaining the predefined execution area of
20 the first operation, the client may firstly obtain the location information of the first virtual object and the predefined execution distance of the first operation, determine area points within the virtual environment, in which the distance between an area point and the first virtual object is less than, or equal to the predefined execution distance. A scope consisting of the area points is the predefined execution area of the first operation.

25 In another possible implementation, the first operation has multiple predefined execution distances. At this time, the distance between a boundary point of the predefined execution area and the first virtual object is not equal. When obtaining the predefined execution area, the client may firstly obtain the location information of the first virtual object and multiple predefined execution distances of the first operation. The client may

take the standing direction of the first virtual object as a reference, determine area points within the virtual environment from various directions of the virtual environment, based on the location information, in which the distance between an area point and the first virtual object is less than, or equal to a corresponding predefined execution distance. A scope consisting of the area points is the predefined execution area of the first operation. The standing direction of the first virtual object may be the real-time moving direction of the first virtual object.

In an example, where the predefined execution area is a rectangle, in conjunction with FIG.4, in the virtual environment, the standing direction of the first virtual object 41 is a first direction 42. When obtaining the predefined execution area 43 of the first operation, the client may obtain the location information of the first virtual object 41 and multiple predefined execution distances of the first operation, take the first direction 42 as a reference, obtain multiple boundary points 44 of the predefined execution area, and then, determine the predefined execution area 43 of the first operation.

It should be noted that, in the embodiments of the application, after displaying a user interface, the client may obtain the predefined execution area of the first operation in real time, and update the predefined execution area, based on the change of the location information of the first virtual object. Alternatively, after receiving the trigger instruction of the first operation, the client may obtain the predefined execution area of the first operation, which is not limited by the embodiments of the application.

In block 303, obtain a visible boundary scope of the virtual environment.

The visible boundary scope refers to a display scope of the user interface. The display scope may be a scope obtained by the virtual camera from the virtual environment. That is, the virtual environment in foregoing visible boundary scope is visible on the user interface, e.g., virtual objects that reside in the visible boundary scope are visible on the user interface. In the embodiments of the application, the client may determine a display scope of the virtual camera, based on screen scope of the client, and then, obtain the visible boundary scope of the virtual environment.

Due to the shooting angle of the virtual camera, the visible boundary scope is not the

same as the screen scope of the client. Exemplarily, in conjunction with FIG.5, a virtual camera 51 may obtain virtual items in a virtual environment, and map the virtual items to a screen scope 52 of the client. Due to the shooting angle of the virtual camera 51, a first virtual object 53 and a solid line portion of a second virtual object 54 may be displayed in the screen scope 52. That is, a dashed scope 55 (i.e., slash coverage area) in FIG.5 is the visible boundary scope.

It should be noted that, the embodiments of the application do not limit the chronological order between the block of obtaining the predefined execution area by the client, and the block of obtaining the visible boundary scope by the client. The client may firstly obtain the predefined execution area, and then, obtain the visible boundary scope. Alternatively, the client may firstly obtain the visible boundary scope, and then, obtain the predefined execution area. Of course, the client may also simultaneously obtain the predefined execution area and the visible boundary scope, which are not limited by the embodiments of the application.

In block 304, determine, based on the predefined execution area and visible boundary scope, an effective execution area of the first operation in the virtual environment.

The effective execution area refers to an action scope of the first operation in an actual operation process. The effective action scope includes the action target of the first operation. The first operation may only affect virtual items, or virtual objects within the effective execution area of the virtual environment, and may not affect virtual items, or virtual objects outside the effective execution area. After obtaining the predefined execution area of foregoing first operation and the visible boundary scope of the virtual environment, the client may determine, based on the predefined execution area and the visible boundary scope, the effective execution area of the first operation in the virtual environment.

In the embodiments of the application, the client may identify the intersection area of the predefined execution area and the visible boundary scope as the effective execution area of the first operation in the virtual environment. Exemplarily, in conjunction with

FIG.6, a predefined execution area 61 of the first operation is a circle scope. A visible boundary scope 62 of the virtual environment is a rectangle scope. The client identifies an intersection area (solid-line scope area) of the predefined execution area 61 and the visible boundary scope 62 as the effective execution area of the first operation.

5 In block 305, determine, based on the effective execution area, a target virtual object.

The target virtual object refers to the action object of foregoing first operation. In the embodiments of the application, after obtaining foregoing effective execution area, the client may determine the target virtual object corresponding to the first operation, based on the effective execution area.

10 In a possible implementation, in order to reduce the response duration of the first operation and improve users' execution experience, after displaying the user interface, the client may obtain the effective execution area of the first operation in real time, determine the target virtual object corresponding to the first operation based on the effective execution area, and update foregoing effective execution area and the target virtual object.

15 Subsequently, after receiving the trigger instruction of foregoing first operation, the client may quickly determine the target virtual object corresponding to the first operation, and control the first virtual object to perform the first operation.

In another possible implementation, in order to reduce processing overhead of a terminal, after obtaining the trigger instruction of the first operation, the client may obtain
20 the effective execution area of the first operation, based on the trigger instruction of the first operation, determine the target virtual object corresponding to the first operation based on the effective execution area, and then, control the first virtual object to perform the first operation.

In view of above, in the technical solutions provided by the embodiments of the
25 application, the effective execution area of the first operation is determined, based on the predefined execution area of the first operation and the visible boundary scope of the virtual environment. The target virtual object corresponding to the first operation is determined, based on the effective execution area, such that the target virtual object is located in the visible boundary scope, thereby guaranteeing that the action object of the

first operation is within the visible scope of the user, avoiding an operation misjudgment resulted from the following scene, i.e., the action object is within an invisible scope of the user, and improving the accuracy for selecting the action target of the first operation.

In addition, identify the intersection area of the predefined execution area and the visible boundary scope as the effective execution area of the first operation, thereby effectively ensuring that the action target of the first operation is not only in the execution area of the first operation, but also in the visible scope of the user.

The obtaining of the visible boundary scope is introduced in the following. In an exemplary embodiment, foregoing block 303 includes the following blocks.

10 Process a three-dimensional visual representation in the virtual environment, and obtain a two-dimensional visual representation of the virtual environment by the processing of the three-dimensional visual representation.

The three-dimensional visual representation refers to a running visual representation of the virtual environment. In the running visual representation, the first virtual object interacts with the virtual environment. The two-dimensional visual representation refers to a picture display diagram of the virtual object. The picture display diagram may be displayed on the user interface of the client.

In the embodiments of the application, when obtaining the visible boundary scope, the client may process the three-dimensional visual representation, and obtain the two-dimensional visual representation in the virtual environment by the processing of the three-dimensional visual representation.

2. Obtain, from the two-dimensional visual representation of the virtual environment, coordinates of feature points of the visible boundary scope within the virtual environment.

25 The feature points are configured to indicate a specific scope of the visible boundary scope. The feature points may be boundary points of the visible boundary scope, e.g., vertexes of the visible boundary scope. It should be noted that, the number of the feature points may be any value, which are not limited by the embodiments of the application.

After obtaining foregoing two-dimensional visual representation of the virtual

environment, the client may obtain, from the two-dimensional visual representation, coordinates of feature points of visible boundary scope within the virtual environment, and then, obtain the visible boundary scope. In the embodiments of the application, the client may obtain, based on parameters of the virtual camera, the coordinates of feature points of the visible boundary scope within the virtual environment. The parameters include a location parameter and a rotation parameter. The location parameter is configured to determine the location of the virtual camera in the virtual environment. The rotation parameter is configured to determine a shooting angle value of the virtual camera in the virtual environment. The client may obtain the shooting angle value of the virtual camera, based on the rotation angle and shooting angle of the virtual camera.

It should be noted that, in the embodiments of the application, since a different client corresponds to a different screen, the client may adjust, based on screen parameters, parameters of the virtual camera, obtain the location parameter and rotation parameter of the virtual camera by the process of the adjustments. And then, the client obtains, based on the location parameter and rotation parameter, the coordinates of feature points of the visible boundary scope within the virtual environment, such that the display screen of the virtual environment obtained by the virtual camera is matched with the user interface of the client. The screen parameters include a screen size and a screen resolution, that is, the client may adaptively adjust the location parameter and rotation parameter of the virtual camera, based on the screen size and screen resolution. Subsequently, the display screen of the virtual environment obtained by the virtual camera may adapt to the screen size and screen resolution of the client.

3. Obtain, based on the coordinates of feature points in the virtual environment, the visible boundary scope.

After obtaining foregoing coordinates of feature points in the virtual environment, the client may obtain, based on the coordinates of feature points in the virtual environment, the visible boundary scope. For example, according to the shape of the visible boundary scope, the client may connect the feature points, and obtain foregoing visible boundary scope.

Exemplarily, assume that the virtual environment includes a three-dimensional coordinate system, the x-axis and y-axis of the three-dimensional coordinate system are parallel to the parallel plane of the virtual environment. The angle between the x-axis and the y-axis is 90° . The z-axis is perpendicular to the parallel plane of the virtual environment. The x-axis, y-axis and z-axis intersect at a point O. Subsequently, the location of the virtual camera in the virtual environment is shown in FIG.7. The coordinates of the virtual camera are (x_1, y_1, z_1) . From the y-axis and z-axis, the value range $z_{min} \sim z_{max}$ of z-axis in the visible boundary scope is obtained. As shown in FIG.7, the rotation angle of the virtual camera on the x-axis is CA, and the shooting angle is FA. The client may determine the rotation angle value of the virtual camera is $CA-FA/2 \sim CA+FA/2$, based on the rotation angle CA and the shooting angle FA. Furthermore, the value range $z_{min} \sim z_{max}$ of z-axis in the visible boundary scope is:

$$z_{min} = z_1 + y_1 * \tan(CA - FA/2);$$

$$z_{max} = z_1 + y_1 * \tan(CA + FA/2) ;$$

Furthermore, from the x-axis and z-axis, the client obtains the vertex coordinates $(x_{TopMin}, z_{max}), (x_{TopMax}, z_{max}), (x_{BotMin}, z_{min}), (x_{BotMax}, z_{min})$ of the visible boundary scope. (x_{TopMin}, z_{max}) is the upper left vertex of the visible boundary scope. (x_{TopMax}, z_{max}) is the upper right vertex of the visible boundary scope. (x_{BotMin}, z_{min}) is the bottom left vertex of the visible boundary scope. (x_{BotMax}, z_{min}) is the bottom right vertex of the visible boundary scope. According to geometric principles, it can be seen that $x_{TopMin}, x_{TopMax}, x_{BotMin}$ and x_{BotMax} are respectively as follows:

$$x_{TopMin} = x_1 - \left(\frac{y_1 / \sin(CA - \frac{FA}{2})}{\tan(CAA)} \right);$$

$$x_{TopMax} = x_1 + \left(\frac{y_1 / \sin(CA - \frac{FA}{2})}{\tan(CAA)} \right);$$

$$x_{BotMin} = x_1 - \left(\frac{y_1 / \sin(CA + \frac{FA}{2})}{\tan(CAA)} \right);$$

$$x_{BotMax} = x_1 + \left(\frac{y_1 / \sin(CA + \frac{FA}{2})}{\tan(CAA)} \right);$$

CAA refers to the horizontal opening angle of the virtual camera.

Since the virtual environment is displayed on the user interface in the form of

two-dimensional visual representation, and the above-mentioned y-axis is perpendicular to the screen inward in the actual coordinate direction of the client, in the two-dimensional visual representation, vertex coordinates of the visible boundary scope are $(x_{TopMin}, 0, z_{max})$, $(x_{TopMax}, 0, z_{max})$, $(x_{BotMin}, 0, z_{min})$ and $(x_{BotMax}, 0, z_{min})$.

- 5 After obtaining the vertex coordinates $(x_{TopMin}, 0, z_{max})$, $(x_{TopMax}, 0, z_{max})$, $(x_{BotMin}, 0, z_{min})$ and $(x_{BotMax}, 0, z_{min})$, the client may connect these vertex coordinates, and then obtain foregoing visible boundary scope. At this time, the visible boundary scope is a trapezoidal scope 71.

It should be noted that, foregoing block of obtaining the visible boundary scope may
10 be performed by a certain computer program module in an application. The computer program module may exist as a plug-in in the application, such as a screen margin filter.

Please refer to FIG.8, FIG.8 is a flowchart illustrating a method for selecting a virtual object on a user interface of a terminal, in accordance with another embodiment of the application. The method may be applied to the terminal, e.g., the execution entity of
15 each block may be the terminal 10 (referred to as "client" in the following) in the APP running environment shown in FIG.1. The method may include several blocks (801~806) as follows.

In block 801, display a user interface.

In block 802, obtain a predefined execution area of a first operation of a first virtual
20 object within a virtual environment.

In block 803, obtain a visible boundary scope of the virtual environment.

In block 804, determine, based on the predefined execution area and the visible boundary scope, an effective execution area of the first operation in the virtual environment.

25 Foregoing blocks 801-804 are the same as blocks 301-304 in the embodiment illustrated with FIG.3, please refer to the embodiment illustrated with FIG.3, which are not repeated here.

In block 805, determine a second virtual object in the effective execution area as a candidate virtual object.

30 The second virtual object refers to a virtual object controlled by the user, or another

user in the application. The candidate virtual object refers to a candidate action target of foregoing first operation. After obtaining foregoing effective execution area, the client may take a virtual object in the effective execution area as the second virtual object. The second virtual object includes a virtual object in the same camp with the first virtual object, and may also include a virtual object in a different camp with the first virtual object.

In a possible implementation, after obtaining foregoing effective execution area, the client may compare coordinates of boundary points of the effective execution area with location coordinates of the virtual object, and then, obtain the second virtual object in the effective execution area.

In another possible implementation, in order to reduce processing overheads of a terminal, the client may directly determine the second virtual object in the effective execution area, based on the predefined execution area and the visible boundary scope. After obtaining the predefined execution area, the client may obtain the second virtual object in the predefined execution area, and obtain the location information of the second virtual object. Furthermore, the client determines whether the second virtual object meets a requirement, based on the location information. The requirement is a judgment requirement used to determine whether the second virtual object is within the effective execution area. If the location information of the second virtual object meets the requirement, the second virtual object is located within the effective execution area. If the location information of the second virtual object does not meet the requirement, the second virtual object is not located within the effective execution area.

Foregoing requirement may include a first requirement and a second requirement. The visible boundary scope is a trapezoidal area enclosed by a first edge, a second edge, a third edge and a fourth edge. Meanwhile, the first edge is parallel to the third edge. After obtaining the location information of the second virtual object within the predefined execution area, the client may analyze and detect the location information. If the location information of foregoing second virtual object is located between the first edge and the third edge, the client determines that the second virtual object meets the first requirement.

If the location information of foregoing second virtual object is located between the second edge and the fourth edge, the client determines that the second virtual object meets the second requirement. When the second virtual object meets the first requirement and the second requirement, the client determines that the second virtual object is located within the effective execution area.

Exemplarily, in conjunction with FIG.7, the visible boundary scope is a trapezoidal scope 71. Vertex coordinates of the trapezoidal scope are $(x_{TopMin}, 0, z_{max})$, $(x_{TopMax}, 0, z_{max})$, $(x_{BotMin}, 0, z_{min})$ and $(x_{BotMax}, 0, z_{min})$. Assume that the coordinates of the second virtual object are (h_x, h_y, h_z) , foregoing first requirement is:

$$z_{min} < h_z < z_{max}.$$

If the location information of the second virtual object meets foregoing first requirement, it is determined that the second virtual object is located between mutually paralleled first edge and third edge in the trapezoidal scope 71.

Foregoing second requirement is:

$$h_x - x_{BotMax} - \left(\frac{(z_{max} - z_{min})}{(x_{TopMax} - x_{BotMax})} \right) * (h_y - z_{min}) > 0;$$

$$\text{and,}$$

$$h_x - x_{BotMin} - \left(\frac{(z_{max} - z_{min})}{(x_{TopMin} - x_{BotMin})} \right) * (h_y - z_{min}) < 0;$$

If the location information of the second virtual object meets foregoing second requirement, it is determined that the second virtual object is located between the second edge and fourth edge in the trapezoidal scope 71.

When the location information of the second virtual object meets the first and second requirements, it is determined that the second virtual object is located within the effective execution area.

The client may select at least one candidate virtual object from multiple second virtual objects, based on operation attributes of foregoing first operation. The operation attributes include an attack attribute and a gain attribute. The attack attribute refers to that foregoing first operation may reduce the attribute value of other virtual objects. The gain attribute refers to that foregoing first operation may increase the attribute value of other

virtual objects. Foregoing attribute value may include a life value, a defense value, or an attack speed, etc., which are not limited by the embodiments of the application. If foregoing first operation is identified as an attack operation, the client identifies a virtual object, which belongs to a different camp from the first virtual object, in the effective execution area as the candidate virtual objects. If foregoing first operation is identified as a gain operation, the client identifies a virtual object, which belongs to the same camp with the first virtual object, in the effective execution area as the candidate virtual objects.

In block 806, select a target virtual object from one or more candidate virtual objects, based on an object selection criteria.

The object selection criteria refers to a selection method of an action target corresponding to the first operation. The object selection criteria includes an action scope of the first operation and a target selection index. The action scope of the first operation is configured to indicate an effect scope of foregoing first operation. The effect scope may be represented by a scope with a certain area and shape, or by the number of target virtual objects. The target selection index is configured to indicate a selection criteria of the target virtual object. The selection criteria may be an attribute value of the second virtual object. For example, the selection criteria is a life value of the second virtual object. Subsequently, the client may select a second candidate virtual object with the smallest life value in the effective execution area as the target virtual object.

In the embodiments of the application, after obtaining foregoing at least one candidate virtual object, the client may select the target virtual object from the at least one candidate virtual object, based on the object selection criteria of the first operation. A different operation corresponds to a different object selection criteria.

It should be noted that, foregoing object selection criteria may be a rule preset by a designer, and may also be a rule that is flexibly changed this time according to the application. For example, the preset rule may change, based on an interactive operation or attribute value of the first virtual object. For example, when the usage count of the first operation of the first virtual object is different, foregoing object selection criteria may be different. For another example, when the attack power of the first virtual object is

different, foregoing object selection criteria may be different, which is not limited by the embodiments of the application.

In view of above, in the technical solutions provided by the embodiments of the application, the target virtual object corresponding to the first operation is selected from the effective execution area, thereby improving the accuracy for selecting the target virtual object on a user interface of a terminal. The target virtual object is selected based on the object selection criteria, thereby the selection of the target virtual object is more flexible.

In addition, a different target virtual object is determined, based on different effects of the first operation, such that the selection of the target virtual object is more flexible.

In addition, in conjunction with FIG.9, a complete introduction to the application is provided.

In block 901, a client displays a user interface.

In block 902, the client obtains relevant parameters about a predefined execution area of a first operation of a first virtual object, in which the relevant parameters of the predefined execution area include location information of the first virtual object and a predefined execution distance of the first operation.

In block 903, the client obtains the predefined execution area of the first operation, based on the relevant parameters of the predefined execution area.

In block 904, the client obtains location information of a second virtual object within the predefined execution area.

In block 905, the client obtains a visible boundary scope.

In block 906, the client determines whether the second virtual object that resides in the predefined execution area meets a requirement. If the second virtual object that reside in the predefined execution area meets the requirement, the client determines that the second virtual object is located in an effective execution area, and executes block 907. If the second virtual object that resides in the predefined execution area does not meet the requirement, the client determines that the second virtual object is not located within the effective execution area, and terminates the process.

In block 907, the client selects a virtual object from multiple second virtual objects within the effective execution area, based on operation attributes of the first operation, and takes the selected virtual object as a candidate virtual object, in which the selected virtual object belongs to the same camp with the first virtual object, or belongs to a different camp from the first virtual object.

In block 908, the client selects a target virtual object from one or more candidate virtual objects, based on an object selection criteria, and takes the selected target virtual object as an action target of the first operation.

The following are apparatus embodiments of the application, which may be configured to execute method embodiments of the application. For details not disclosed in the apparatus embodiments of the application, please refer to method embodiments of the application.

Please refer to FIG.10, FIG.10 is a block diagram of an apparatus for selecting a virtual object on a user interface of a terminal, in accordance with an embodiment of the application. The apparatus has the function of realizing the selection method of above virtual object. Such function may be implemented by hardware, and may also be implemented by hardware executing corresponding software. The apparatus may be the terminal, and may also be set in the terminal. An apparatus 1000 may include an interface display module 1010, an area obtaining module 1020, a boundary obtaining module 1030, an area determining module 1040 and an object determining module 1050.

The interface display module 1010 is configured to display a user interface. The user interface includes a display screen corresponding to a virtual environment, and further includes a first virtual object located in the virtual environment.

The area obtaining module 1020 is configured to obtain a predefined execution area of a first operation of the first virtual object within the virtual environment.

The boundary obtaining module 1030 is configured to obtain a visible boundary scope of the virtual environment. Virtual objects that reside in the visible boundary scope are visible on the user interface.

The area determining module 1040 is configured to determine, based on the

predefined execution area and the visible boundary scope, an effective execution area of the first operation within the virtual environment.

The object determining module 1050 is configured to determine, based on the effective execution area, a target virtual object.

5 In an exemplary embodiment, the boundary obtaining module 1030 includes a two-dimensional obtaining unit 1031, a coordinate obtaining unit 1032 and a boundary obtaining unit 1033.

10 The two-dimensional obtaining unit 1031 is configured to process a three-dimensional visual representation in the virtual environment, and obtain a two-dimensional visual representation of the virtual environment by the processing of the three-dimensional visual representation.

The coordinate obtaining unit 1032 is configured to obtain, from the two-dimensional visual representation of the virtual environment, coordinates of feature points of the visible boundary scope within the virtual environment.

15 The boundary obtaining unit 1033 is configured to obtain, based on the coordinates of the feature points in the virtual environment, the visible boundary scope.

In an exemplary embodiment, the boundary obtaining unit 1033 includes a coordinate obtaining subunit.

20 The coordinate obtaining subunit is configured to obtain, based on parameters of a virtual camera, coordinates of the feature points of the visible boundary scope within the virtual environment. The parameters include a location parameter and a rotation parameter. The location parameter is configured to determine the location of the virtual camera in the virtual environment. The rotation parameter is configured to determine a shooting angle value of the virtual camera in the virtual environment.

25 In an exemplary embodiment, the coordinate obtaining subunit is configured to adjust, based on screen parameters, the parameters of the virtual camera, and obtain the location parameter and the rotation parameter of the virtual camera, by the process of the adjustments. The screen parameters include parameters of screen size and screen resolution. The coordinate obtaining subunit is configured to obtain, based on the location

parameter and the rotation parameter, the coordinates of the feature points of the visible boundary scope within the virtual environment.

In an exemplary embodiment, the area obtaining module 1020 is configured to obtain the location information of the first virtual object in the virtual environment, and
5 determine, based on the location information and the predefined execution distance of the first operation, the predefined execution area of the first operation in the virtual environment.

In an exemplary embodiment, the object determining module 1050 includes a candidate determining unit 1051 and a target determining unit 1052.

10 The candidate determining unit 1051 is configured to determine a second virtual object within the effective execution area as a candidate virtual object.

The target determining unit 1052 is configured to select the target virtual object from one or more candidate virtual objects, based on an object selection criteria.

In an exemplary embodiment, the visible boundary scope is a trapezoidal area
15 enclosed by a first edge, a second edge, a third edge and a fourth edge. The first edge is parallel to the third edge. As shown in FIG.11, the apparatus 1000 further includes a location determining module 1060.

The location determining module 1060 is configured to obtain location information of the second virtual object in the predefined execution area. When the location
20 information of the second virtual object is located between the first edge and the third edge, the location determining module 1060 is configured to determine that the second virtual object meets a first requirement. When the location information of the second virtual object is located between the second edge and the fourth edge, the location determining module 1060 is configured to determine that the second virtual object meets
25 a second requirement. When the second virtual object meets the first requirement and the second requirement, the location determining module 1060 is configured to determine that the second virtual object is located in the effective execution area.

In an exemplary embodiment, when the first operation is identified as an attack operation, the candidate determining unit 1051 is configured to determine a virtual object,

which belongs to a different camp from the first virtual object, in the effective execution area as the candidate virtual object. Alternatively, when the first operation is identified as a gain operation, the candidate determining unit 1051 is configured to determine a virtual object, which belongs to the same camp as the first virtual object, in the effective
5 execution area as the candidate virtual object.

In view of above, in the technical solutions provided by the embodiments of the application, the effective execution area of the first operation is determined, based on the predefined execution area of the first operation and the visible boundary scope of the virtual environment. The target virtual object corresponding to the first operation is
10 determined, based on the effective execution area. Subsequently, the target virtual object is located within the visible boundary scope, thereby ensuring that the action target of the first operation is located within the visible scope of the user, avoiding an operation misjudgment resulted from the following scene, i.e., the action target is located in the invisible scope of the user, and improving the accuracy for selecting the action target of
15 the first operation.

It should be noted that, for the apparatus provided by foregoing embodiment, when implementing functions thereof, only the division of foregoing functional modules is taken as an example. In practical applications, foregoing functions may be allocated to different functional modules to be completed, according to needs. That is, an internal
20 structure of the device is divided into different functional modules, so as to complete all of or some functions described above. In addition, the apparatus provided by foregoing embodiments and method embodiments belong to the same concept. The specific implementation process may refer to method embodiments, which are not repeated here.

Please refer to FIG.12, FIG.12 is a block diagram illustrating structure of a terminal
25 1200, in accordance with an embodiment of the application. The terminal 1200 may be an electronic device, such as a mobile phone, a tablet computer, a game host, an ebook reader, a multimedia playback device, a wearable device and a Personal Computer (PC). The terminal is configured to implement the method for selecting a virtual object provided by foregoing embodiment. The terminal may be the terminal 10 in the game

running environment illustrated with FIG.1.

Generally, the terminal 1200 includes a processor 1201 and a memory 1202.

The processor 1201 may include one or more processing core, e.g., a 4-core processor, a 8-core processor, and so on. The processor 1201 may be implemented by at least one hardware form, e.g., Digital Signal Processing (DSP), Field Programmable Gate Array (FPGA), Programmable Logic Array (PLA). The processor 1201 may also include a main processor and a coprocessor. The main processor is a processor in charge of processing data in awake state, which may also be referred to as a Central Processing Unit (CPU). The coprocessor is a low-power processor configured to process data in the standby state. In some embodiments, the processor 1201 may be integrated with a Graphics Processing Unit (GPU). The GPU is configured to render and draw contents needing to be displayed on the display. In some embodiments, the processor 1201 may also include an Artificial Intelligence (AI) processor. The AI processor is configured to process computing operations related with machine learning.

The memory 1202 may include one or more computer readable storage medium. The computer readable storage medium may be non-transitory. The memory 1202 may also include a high-speed Random Access Memory (RAM), and a non-transitory memory, e.g., one or more disk storage device, flash storage device. In some embodiments, the non-transitory computer readable storage medium in the memory 1202 is configured to store at least one instruction, at least one program, code set or instruction set, in which the at least one instruction, at least one program, code set or instruction set are configured and processed by one or more processor, so as to implement foregoing method for selecting a virtual object.

In some embodiments, the terminal 1200 may also include a peripheral device interface 1203 and at least one peripheral device. The processor 1201, the memory 1202 and the peripheral device interface 1203 may be connected through a bus or signal line. Each peripheral device may be connected with the peripheral device interface 1203 through the bus, the signal line or a circuit board. Specifically, the peripheral device includes at least one of a Radio Frequency (RF) circuit, a display (e.g., a touch screen)

1205, a camera component 1206, an audio circuit 1207, a locating component 1208, a power supply 1209.

Persons having ordinary skill in the art may understand that, the structure shown in FIG.12 does not limit the terminal 1200. The terminal 1200 may include more or less components compared with that shown in FIG.12, or combine some components, or adopt a different component layout.

In an exemplary embodiment, a computer readable storage medium is also provided. The storage medium stores at least one instruction, at least one program, code set or instruction set, in which when the at least one instruction, the at least one program, the code set or instruction set are executed by the processor, the above-mentioned method for selecting a virtual object is implemented.

The computer readable storage medium may include a Read Only Memory (ROM), a Random Access Memory (RAM), a Solid State Drives (SSD) or a Compact Disk (CD), and so on. The RAM may include a Resistance Random Access Memory (ReRAM) and a Dynamic Random Access Memory (DRAM).

In an exemplary embodiment, a computer program product is also provided. When the computer program product is executed by the processor, foregoing method for selecting a virtual object is achieved.

It should be understood that, “multiple” mentioned in the application refers to two or more. “And/or” describes an associated relationship of associated objects, which indicates that there may be three relationships, e.g., A and/or B may indicate three situations, A exists independently, A and B simultaneously exist, B exists independently. The character “/” generally represents that there is a “or” relationship between the former and subsequent associated objects. In addition, the block numbers described in the application only exemplarily illustrate one possible execution order among blocks. In some other embodiments, foregoing blocks may be performed not according to the sequence of the numbers, e.g., two blocks with different numbers are performed simultaneously, or two blocks with different numbers are performed according to an inverse order illustrated in the figure, which are not limited by the embodiments of the application.

The foregoing are only exemplary embodiments of the application, which are not for use in limiting the application. Any modifications, equivalent substitutions, or improvements made within the spirit and principle of the application should be covered by the protection scope of the application.

Claims

1. A method for selecting a virtual object on a user interface of a terminal, performed by the terminal, comprising:

displaying a user interface, wherein the user interface comprises a display screen
5 corresponding to a virtual environment, and further comprises a first virtual object located in the virtual environment;

obtaining a predefined execution area of a first operation of the first virtual object within the virtual environment;

obtaining a visible boundary scope of the virtual environment, wherein virtual objects
10 that reside in the visible boundary scope are visible on the user interface;

when a portion of the predefined execution area is outside the visible boundary scope, identifying an intersection area of the predefined execution area and the visible boundary scope as an effective execution area of the first operation; and

determining, based on the effective execution area, a target virtual object.

15 2. The method according to claim 1, wherein the obtaining the visible boundary scope of the virtual environment comprises:

processing a three-dimensional visual representation of the virtual environment;
obtaining a two-dimensional visual representation of the virtual environment by the processing of the three-dimensional visual representation;

20 obtaining, from the two-dimensional visual representation, coordinates of feature points of the visible boundary scope within the virtual environment; and

obtaining, based on the coordinates of the feature points, the visible boundary scope.

3. The method according to claim 2, wherein the obtaining the coordinates of the feature points of the visible boundary scope within the virtual environment comprises:

25 obtaining, based on parameters of a virtual camera, coordinates of the feature points of the visible boundary scope within the virtual environment;

wherein the parameters comprise a location parameter and a rotation parameter, the location parameter is configured to determine the location of the virtual camera in the virtual environment, the rotation parameter is configured to determine a shooting angle

value of the virtual camera in the virtual environment.

4. The method according to claim 3, wherein the obtaining the coordinates of the feature points of the visible boundary scope within the virtual environment, based on the parameters of the virtual camera, comprises:

- 5 adjusting, based on screen parameters, the parameters of the virtual camera;
- obtaining the location parameter and the rotation parameter of the virtual camera, by the process of the adjustments, wherein the screen parameters comprise parameters of screen size and screen resolution; and
- obtaining, based on the location parameter and the rotation parameter, the coordinates
- 10 of the feature points of the visible boundary scope within the virtual environment.

5. The method according to claim 1, wherein the obtaining the predefined execution area of the first operation of the first virtual object in the virtual environment comprises:

- obtaining location information of the first virtual object within the virtual environment;
- and,
- 15 determining, based on the location information and a predefined execution distance of the first operation, the predefined execution area of the first operation in the virtual environment.

6. The method according to claim 1, wherein determining the target virtual object based on the effective execution area comprises:

- 20 determining a second virtual object within the effective execution area as a candidate virtual object;
- selecting the target virtual object from one or more candidate virtual object based on an object selection criteria.

7. The method according to claim 6, wherein the visible boundary scope is a

25 trapezoidal area enclosed by a first edge, a second edge, a third edge and a fourth edge, the first edge is parallel to the third edge, and the method further comprises:

 obtaining location information of the second virtual object in the predefined execution area;

 when the location information of the second virtual object is located between the first

edge and the third edge, determining that the second virtual object meets a first requirement;

when the location information of the second virtual object is between the second edge and the fourth edge, determining that the second virtual object meets a second requirement;

when the second virtual object meets the first requirement and the second requirement,
5 determining that the second virtual object is located in the effective execution area.

8. The method according to claim 6, wherein the determining the second virtual object in the effective execution area as the candidate virtual object comprises:

when the first operation is identified as an attack operation, identifying a virtual object, which belongs to a different camp from the first virtual object, in the effective execution
10 area as the candidate virtual object.

9. The method according to claim 6, wherein determining the second virtual object in the effective execution area as the candidate virtual object comprises:

when the first operation is identified as a gain operation, identifying a virtual object which belongs to the same camp with the first virtual object, in the effective execution area
15 as the candidate virtual object.

10. An apparatus for selecting a virtual object on a user interface of a terminal, comprising:

an interface display module, configured to display a user interface, wherein the user interface comprises a display screen corresponding to a virtual environment, and further
20 comprises a first virtual object located in the virtual environment;

a scope obtaining module, configured to obtain a predefined execution area of a first operation of the first virtual object within the virtual environment;

a boundary obtaining module, configured to obtain a visible boundary scope of the virtual environment, wherein virtual objects that reside in the visible boundary scope are
25 visible on the user interface;

a scope determining module, configured to, when a portion of the predefined execution area is outside the visible boundary scope, identify an intersection area of the predefined execution area and the visible boundary scope as an effective execution area of the first operation; and,

an object determining module, configured to determine, based on the effective execution area, a target virtual object.

11. The apparatus according to claim 10, wherein the boundary obtaining module comprises a two-dimensional obtaining unit, a coordinate obtaining unit and a boundary
5 obtaining unit;

the two-dimensional obtaining unit is configured to process a three-dimensional visual representation of the virtual environment, obtain a two-dimensional visual representation of the virtual environment by the processing of the three-dimensional visual representation;

the coordinate obtaining unit is configured to obtain, from the two-dimensional visual
10 representation of the virtual environment, coordinates of feature points of the visible boundary scope within the virtual environment; and,

the boundary obtaining unit is configured to obtain, based on the coordinates of the feature points within the virtual environment, the visible boundary scope.

12. The apparatus according to claim 11, wherein the boundary obtaining unit
15 comprises a coordinate obtaining subunit,

the coordinate obtaining subunit is configured to obtain, based on parameters of a virtual camera, coordinates of the feature points of the visible boundary scope within the virtual environment,;

wherein the parameters comprise a location parameter and a rotation parameter, the
20 location parameter is configured to determine the location of the virtual camera in the virtual environment, the rotation parameter is configured to determine a shooting angle value of the virtual camera in the virtual environment.

13. A terminal, comprising a processor and a memory, wherein the memory stores at least one instruction, at least one program, a code set or an instruction set, when being
25 loaded and executed by the processor, causing the processor to perform the method for selecting a virtual object according to any of claims 1, 3 to 10.

14. A computer readable storage medium, wherein the computer readable storage medium stores at least one instruction, at least one program, a code set or an instruction set, when being loaded and processed by a processor, causing the processor to perform the

method for selecting a virtual object according to any of claims 1, 3 to 10.