United States Patent Application Publication	20250259398
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	Crispin; Sterling et al.

MAINTAINING AR/VR CONTENT AT A RE-DEFINED POSITION

Abstract

The systems and methods display, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space. The systems and methods receive a request to move the one or more virtual objects to a second virtual coordinate in 3D space. The systems and methods in response to receiving the request, maintain display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.

Inventors: **Crispin; Sterling (Santa Cruz, CA), Stolzenberg; Karen (Venice, CA), Wong; Brian (Murrieta, CA)**

Applicant: **Crispin; Sterling** (Santa Cruz, CA); **Stolzenberg; Karen** (Venice, CA); **Wong; Brian** (Murrieta, CA)

Family ID: **1000007700523**

Appl. No.: **18/436580**

Filed: **February 08, 2024**

Publication Classification

Int. Cl.: **G06T19/00** (20110101); **G06T19/20** (20110101); **G06V40/20** (20220101)

U.S. Cl.:

CPC **G06T19/006** (20130101); **G06T19/20** (20130101); **G06V40/20** (20220101); G06T2219/2004 (20130101)

Background/Summary

FIELD OF USE

[0001] This disclosure relates to augmented reality devices.

BACKGROUND

[0002] Some electronics-enabled eyewear devices, such as so-called smart glasses, allow users to interact with virtual content while a user is engaged in some activity. Users wear the eyewear devices and can view a real-world environment through the eyewear devices while interacting with virtual content that is displayed by the eyewear devices.

Description

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] Various ones of the appended drawings merely illustrate examples of the present disclosure and should not be considered as limiting its scope.

[0004] FIG. **1** is a diagrammatic representation of a networked environment in which the present disclosure may be deployed, in accordance with some examples.

[0005] FIG. **2** is a diagrammatic representation of a messaging system, in accordance with some examples, that has both client-side and server-side functionality.

[0006] FIG. **3** is a diagrammatic representation of a data structure as maintained in a database, in accordance with some examples.

[0007] FIG. **4** is a perspective view of an eyewear device according to an example.

[0008] FIG. **5** is a flowchart showing example operations of the content centering system according to an example.

[0009] FIGS. **6**-**8** are illustrative screens of the content centering system according to examples.

[0010] FIG. **9** is a diagrammatic representation of a machine in the form of a computer system within which a set of instructions may be executed for causing the machine to perform any one or more of the methodologies discussed herein, in accordance with some examples.

[0011] FIG. **10** is a block diagram showing a software architecture within which examples may be implemented.

DETAILED DESCRIPTION

[0012] The description that follows discusses illustrative examples of the disclosure. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide an understanding of various examples of the disclosed subject matter. It will be evident, however, to those skilled in the art, that examples of the disclosed subject matter may be practiced without these specific details. In general, well-known instruction instances, protocols, structures, and techniques are not necessarily shown in detail.

[0013] Typical smart glass platforms allow users to read their text messages within the smart glasses as well as interact with other types of virtual content. Such platforms are configured to display the virtual content in the lenses of the smart glasses. While such systems work well to generally allow users to interact with virtual content, such devices fail to consider movement of the user through the real-world environment in the display of the virtual content. In addition, such virtual content is usually positioned, fixed, in the real world and can be left behind or lost, as the user moves to a different real world position. Even though this is how real world objects behave, in virtual spaces, this can feel inconvenient, confusing, and frustrating.

[0014] Some systems continuously display the virtual content as the user moves around the real-world environment but doing so does not always make sense and can end up inundating the user with too much information making such presentation counterproductive. Namely, such devices fail to selectively determine whether or not to continue to display virtual content within a current view of the real-world environment as the user moves around which, in turn, ends up frustrating the user and ends up distracting the user from activities they are performing. Depending on its

constraints/tuning, this behavior can also be uncomfortable and feel claustrophobic, such as if the virtual content directly **1**:**1** follows the user's head motion. Also if the content **1**:**1** follows the user as they move around the real world, and it is a large size, and if it is not adjustable, the content can also block the user's view, and distract them from real world obstacles/information

[0015] Some smart glass platforms display the virtual content at predefined default positions in three-dimensional (3D) space. These positions are set to be a certain virtual distance from the smart glasses (e.g., the user's face) and at a certain virtual height above the real-world surface being viewed through the smart glasses. The positions are selected by a provider or developer of the AR content to be comfortable for most users. However, some users have different physical reach capabilities (e.g., because they have longer or shorter arms) and so the default positions may not be the most appealing for these users. These users often get frustrated in having to reach too far into virtual space to make certain selections or interact with the virtual content. The typical smart glass platforms fail to provide a means for users to specify or define a comfortable level of distance and height for virtual objects, which causes the users to constantly interact with virtual content in an uncomfortable manner.

[0016] In addition, the typical way of presenting the virtual content in the lenses of the smart glasses ends up consuming a great deal of processing and battery resources. This is because such devices use standard, resource-intensive programming languages and operations to process the virtual content and further use additional resources to generate such virtual content for display. As a result, the battery life of these typical smart glasses is very limited, requiring a user to constantly charge the smart glasses for use, which takes away from the appeal and interest of using the smart glasses.

[0017] The disclosed examples improve the efficiency of using the electronic device by allowing users to re-define where virtual content is positioned relative to a user in virtual 3D space. The disclosed examples provide a solution that improves comfort for the user in navigating virtual content by allowing the user to comfortably reach UI elements with hand gestures. Also, the experience is improved since the user always knows where the UI is positioned in virtual space and so the virtual content cannot be lost as the user moves around. The disclosed examples improve safety of AR devices by providing a reliable way to control inputs for user interfaces that contain critical privacy UI controls like mute/camera off. The disclosed techniques enable the electronic device (AR device) to customize the default display positions for virtual content to each user's needs and based on the physical attributes of each user. This way, the user can comfortably interact with the virtual content using hand gestures or other input modalities. Specifically, the disclosed techniques enable users to refine or adjust the position of their virtual content after the virtual content has been initially defined by the system to follow the AR device position. This adjusted position can then be saved/cached/restored for future sessions. A single predefined position for virtual content following a user can be uncomfortable for hand interactions for different-sized users. Thus, the disclosed examples allow for additional movement, adjustment of virtual content, while keeping the virtual content visible in the field-of-view (FOV) of the user, which can be helpful and increases user comfort in operating these interfaces.

[0018] Namely, the disclosed examples display, on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in 3D space. The disclosed examples receive a request to move the one or more virtual objects to a second virtual coordinate in 3D space. The disclosed examples, in response to receiving the request, maintain a display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment becomes visible on the display of the user device. Specifically, interacting with virtual content while moving around a real-world environment can cause users to lose track of where the virtual content (e.g., virtual menus) is placed. The disclosed examples ensure that the virtual content is maintained in a visible FOV in a comfortable manner so the user can always interact with the virtual content in a non-disruptive manner.

[0019] Because the processor of the eyewear device executes a low-power process to selectively bring into view virtual objects that are displayed within a real-world environment, the battery life of the electronic eyewear device is enhanced. This increases the efficiency, appeal, and utility of electronic eyewear devices. The virtual content can both follow the user and is moveable by the user (e.g., using hand gestures) with constraints to remain visible within the FOV of the user which is useful for user comfort, user convenience and user safety.

Networked Computing Environment

[0020] FIG. **1** is a block diagram showing an example system **100** for exchanging data (e.g., messages and associated content) over a network. The system **100** includes multiple instances of a user device **102**, each of which hosts a number of applications, including an interaction application **104** and other external applications **109** (e.g., third-party applications). Each interaction application **104** is communicatively coupled to other instances of the interaction application **104** (e.g., hosted on respective other user devices **102**, sometimes referred to as user system(s)), a server system **108** and external app(s) servers **110** via a network **112** (e.g., the Internet). An interaction application **104** can also communicate with locally-hosted third-party applications **109** using Applications Program Interfaces (APIs). The system **100** includes an eyewear device **119**, which hosts a content centering system **107**, among other applications. The eyewear device **119** is communicatively coupled to the user device **102** via the network **112** (which may include via a dedicated short-range communication path, such as a Bluetooth™ or WiFi direct connection).

[0021] The content centering system **107** automatically or selectively moves augmented reality or virtual reality content items from one virtual position to another as the user moves around the eyewear device **119**. For example, the user or wearer of the eyewear device **119** may initially be looking at a first portion of a real-world environment (e.g., a first room in a house). The user may provide input (e.g., using a user device **102** or a voice-activated or touch-activated interface of the eyewear device **119**) to launch or access virtual content that includes one or more virtual objects. Specifically, the user can request to access a music or video library. In response, a list of thumbnails or visual indicators of the media assets associated with a music or video library is retrieved and presented within view of the first portion of the real-world environment. Other virtual content can also be presented, such as menu items or any other content item that can be used to control other content items being displayed or accessed.

[0022] In an example, the content centering system **107** assigns a first virtual location to the virtual content. The first virtual location corresponds to a position in three-dimensional (3D) space at which the first portion of the real-world environment exists. The content centering system **107** displays the virtual content at the first virtual location so that the virtual content is visible to the user as the user views the first portion of the real-world environment through lenses of the eyewear device **119** (or other AR display device). In one case, the virtual content includes user interface elements for controlling playback of content. In such cases, the user can interact with the virtual content to playback, stop, skip or control consumption of the media assets. In another case, the virtual content includes user interface elements for generating augmented reality content. In such cases, the user can interact with the virtual content to add augmented reality items or elements to a view of the eyewear device **119** (e.g., the user can add virtual paint to a real-world scene the user is viewing through the eyewear device **119**). Any functionality described herein with respect to the eyewear device **119** can similarly be applied or performed by the user device **102** or other AR/VR device.

[0023] While the media assets are being played back and displayed at the first virtual location, the user can turn their head or walk to another real-world location (e.g., a second portion of the real-world environment, such as another room in the house). The other real-world location may include some or none of the real-world objects that are within view of the first portion of the real-world location. The virtual content that was displayed at the first virtual location is moved to the new location so that the virtual content is persistently displayed for the user to interact with the virtual

content.
[0024] In some cases, the virtual content is initially displayed at an initial or default virtual location. This virtual location can be at a first virtual distance from the user or the eyewear device **119** and at a first virtual height relative to the real-world surface that appears through the lenses of the eyewear device **119**. The eyewear device **119** can receive input from the user that moves the virtual content in 3D space closer or farther from the user or a point (e.g., the lenses) of the eyewear device **119** (e.g., along the z-axis), moves the virtual content up/down relative to the real-world surface (e.g., along the y-axis), and/or moves the virtual content left/right relative to the user or the point of the eyewear device **119**. In such cases, the content centering system **107** computes a virtual offset that represents the new virtual position of the virtual content relative to the user or point on the eyewear device **119**. The content centering system **107** can maintain display of the virtual content at the same virtual offset relative to the user and/or point on the eyewear device **119** as the user moves around to view other portions of the real-world environment. This way, the user can continuously interact with the virtual content as the user moves around and the virtual content is displayed at a convenient location for the user.
[0025] In some cases, the content centering system **107** receives input from the user that drags the virtual content to a new virtual position that is outside of the field-of-view (FOV) of the eyewear device **119**. This can be a point where cameras of the eyewear device **119** no longer have a corresponding real-world coordinate in the images being captured that corresponds to the virtual position to which the virtual content has been moved. The content centering system **107** can receive input that presses or otherwise selects the virtual content and drags the virtual content to the new virtual position. In response to receiving input that releases the virtual content and/or sets the virtual content at the new virtual position, the content centering system **107** automatically re-centers the virtual content at a predetermined or user-specified position that is within the FOV of the eyewear device **119**. This way, if the user moves virtual content out of view, the virtual content is automatically brought back into view. In some cases, the virtual content is brought back into view to a virtual position that is computed based on the previously computed virtual offset relative to the point on the eyewear device **119** or the user.
[0026] In some cases, the content centering system **107** applies the same virtual offset to other virtual content that is retrieved and displayed on the eyewear device **119**. For example, the content centering system **107** can receive input that requests a new menu item to be virtually presented in the lenses of the eyewear device **119**. In response, the content centering system **107** obtains the new menu item and retrieves a default virtual coordinate that is associated with the new menu item. The virtual coordinate is a position in a virtual space frame of reference of the eye wear device **119** that defines the height and distance and left/right position at which to present virtual content. The content centering system **107** can determine whether the default virtual coordinate corresponds to the virtual offset associated with the user. In response to determining that the default virtual coordinate fails to correspond to the virtual offset associated with the user, the content centering system **107** updates or modifies the default virtual coordinate associated with the new menu item to match the virtual offset associated with the user. This way, the new menu item is presented at the same or substantially the same virtual height relative to the real-world surface and/or same virtual distance from the point on the eyewear device **119** as the virtual content that was previously displayed and moved by the user. The content centering system **107** can update the default display positions of all virtual content items or all virtual content items that are of the same or similar type as the virtual content based on the virtual offset that was previously computed.
[0027] The interaction application **104** is able to communicate and exchange data with other interaction applications **104**, the eyewear device **119**, and with the server system **108** via the network **112**. The data exchanged between interaction applications **104**, and between the interaction application **104** and the server system **108**, includes functions (e.g., commands to invoke functions) as well as payload data (e.g., text, audio, video, or other multimedia data).

[0028] The server system **108** provides server-side functionality via the network **112** to an interaction application **104**. While certain functions of the system **100** are described herein as being performed by either an interaction application **104** or by the server system **108**, the location of certain functionality either within the interaction application **104** or the server system **108** may be a design choice. For example, it may be technically preferable to initially deploy certain technology and functionality within the server system **108** but to later migrate this technology and functionality to the interaction application **104** where a user device **102** has sufficient processing capacity.
[0029] The server system **108** supports various services and operations that are provided to the interaction application **104**. Such operations include transmitting data to, receiving data from, and processing data generated by the interaction application **104**. This data may include message content, client device information, geolocation information, media augmentation and overlays, message content persistence conditions, social network information, and live event information, as examples. Data exchanges within the system **100** are invoked and controlled through functions available via user interfaces (UIs) of the interaction application **104**.
[0030] Turning now specifically to the server system **108**, an API server **116** is coupled to, and provides a programmatic interface to, application servers **114**. The application servers **114** are communicatively coupled to a database server **120**, which facilitates access to a database **126** that stores data associated with messages processed by the application servers **114**. Similarly, a web server **128** is coupled to the application servers **114**, and provides web-based interfaces to the application servers **114**. To this end, the web server **128** processes incoming network requests over the Hypertext Transfer Protocol (HTTP) and several other related protocols.
[0031] The Application Program Interface (API) server **116** receives and transmits message data (e.g., commands and message payloads) between the user device **102** and the application servers **114**. Specifically, the Application Program Interface (API) server **116** provides a set of interfaces (e.g., routines and protocols) that can be called or queried by the interaction application **104** in order to invoke functionality of the application servers **114**. The Application Program Interface (API) server **116** exposes various functions supported by the application servers **114**, including account registration, login functionality, the sending of messages, via the application servers **114**, from a particular interaction application **104** to another interaction application **104**, the sending of media files (e.g., images or video) from a interaction application **104** to a interaction application server **118**, and for possible access by another interaction application **104**, the settings of a collection of media data (e.g., story), the retrieval of a list of friends of a user of a user device **102**, the retrieval of such collections, the retrieval of messages and content, the addition and deletion of entities (e.g., friends) to an entity graph (e.g., a social graph), the location of friends within a social graph, and opening an application event (e.g., relating to the interaction application **104**).
[0032] The application servers **114** host a number of server applications and subsystems, including for example an interaction application server **118**, an image processing server **122**, and a social network server **124**. The interaction application server **118** implements a number of message processing technologies and functions, particularly related to the aggregation and other processing of content (e.g., textual and multimedia content) included in messages received from multiple instances of the interaction application **104**. As will be described in further detail, the text and media content from multiple sources may be aggregated into collections of content (e.g., called stories or galleries). These collections are then made available to the interaction application **104**. Other processor- and memory-intensive processing of data may also be performed server-side by the interaction application server **118**, in view of the hardware requirements for such processing.
[0033] The application servers **114** also include an image processing server **122** that is dedicated to performing various image processing operations, typically with respect to images or video within the payload of a message sent from or received at the interaction application server **118**.
[0034] Image processing server **122** is used to implement scan functionality of the augmentation system **208** (FIG. **2**). Scan functionality includes activating and providing one or more augmented

reality experiences on a user device **102** when an image is captured by the user device **102**. Specifically, the interaction application **104** on the user device **102** can be used to activate a camera. The camera displays one or more real-time images or a video to a user along with one or more icons or identifiers of one or more augmented reality experiences. The user can select a given one of the identifiers to launch the corresponding augmented reality experience. Launching the augmented reality experience includes obtaining one or more augmented reality items associated with the augmented reality experience and overlaying the augmented reality items on top of the images or video being presented.

[0035] The social network server **124** supports various social networking functions and services and makes these functions and services available to the interaction application server **118**. To this end, the social network server **124** maintains and accesses an entity graph **308** (as shown in FIG. **3**) within the database **126**. Examples of functions and services supported by the social network server **124** include the identification of other users of the system **100** with which a particular user has relationships or is "following," and also the identification of other entities and interests of a particular user.

[0036] Returning to the interaction application **104**, features and functions of an external resource (e.g., a third-party application **109** or applet) are made available to a user via an interface of the interaction application **104**. The interaction application **104** receives a user selection of an option to launch or access features of an external resource (e.g., a third-party resource), such as external apps **109**. The external resource may be a third-party application (external apps **109**) installed on the user device **102** (e.g., a "native app"), or a small-scale version of the third-party application (e.g., an "applet") that is hosted on the user device **102** or remote of the user device **102** (e.g., on third-party servers **110**). The small-scale version of the third-party application includes a subset of features and functions of the third-party application (e.g., the full-scale, native version of the third-party standalone application) and is implemented using a markup-language document. In one example, the small-scale version of the third-party application (e.g., an "applet") is a web-based, markup-language version of the third-party application and is embedded in the interaction application **104**. In addition to using markup-language documents (e.g., a .*ml file), an applet may incorporate a scripting language (e.g., a .*js file or a .json file) and a style sheet (e.g., a .*ss file).

[0037] In response to receiving a user selection of the option to launch or access features of the external resource (external app **109**), the interaction application **104** determines whether the selected external resource is a web-based external resource or a locally-installed external application. In some cases, external applications **109** that are locally installed on the user device **102** can be launched independently of and separately from the interaction application **104**, such as by selecting an icon, corresponding to the external application **109**, on a home screen of the user device **102**. Small-scale versions of such external applications can be launched or accessed via the interaction application **104** and, in some examples, no or limited portions of the small-scale external application can be accessed outside of the interaction application **104**. The small-scale external application can be launched by the interaction application **104** receiving, from an external app(s) server **110**, a markup-language document associated with the small-scale external application and processing such a document.

[0038] In response to determining that the external resource is a locally-installed external application **109**, the interaction application **104** instructs the user device **102** to launch the external application **109** by executing locally-stored code corresponding to the external application **109**. In response to determining that the external resource is a web-based resource, the interaction application **104** communicates with the external app(s) servers **110** to obtain a markup-language document corresponding to the selected resource. The interaction application **104** then processes the obtained markup-language document to present the web-based external resource within a user interface of the interaction application **104**.

[0039] The interaction application **104** can notify a user of the user device **102**, or other users

related to such a user (e.g., “friends”), of activity taking place in one or more external resources. For example, the interaction application **104** can provide participants in a conversation (e.g., a chat session) in the interaction application **104** with notifications relating to the current or recent use of an external resource by one or more members of a group of users. One or more users can be invited to join an active external resource or to launch a recently-used but currently inactive (in the group of friends) external resource. The external resource can provide participants in a conversation, each using a respective interaction application **104**, with the ability to share an item, status, state, or location in an external resource with one or more members of a group of users into a chat session. The shared item may be an interactive chat card with which members of the chat can interact, for example, to launch the corresponding external resource, view specific information within the external resource, or take the member of the chat to a specific location or state within the external resource. Within a given external resource, response messages can be sent to users on the interaction application **104**. The external resource can selectively include different media items in the responses, based on a current context of the external resource.

[0040] The interaction application **104** can present a list of the available external resources (e.g., third-party or external applications **109** or applets) to a user to launch or access a given external resource. This list can be presented in a context-sensitive menu. For example, the icons representing different ones of the external application **109** (or applets) can vary based on how the menu is launched by the user (e.g., from a conversation interface or from a non-conversation interface).

System Architecture

[0041] FIG. **2** is a block diagram illustrating further details regarding the system **100**, according to some examples. Specifically, the system **100** is shown to include the interaction application **104** and the application servers **114**. The system **100** embodies a number of subsystems, which are supported on the client side by the interaction application **104** and on the server side by the application servers **114**. These subsystems include, for example, an ephemeral timer system **202**, a collection management system **204**, an augmentation system **208**, a map system **210**, a game system **212**, and an external resource system **220**.

[0042] The ephemeral timer system **202** is responsible for enforcing the temporary or time-limited access to content by the interaction application **104** and the interaction application server **118**. The ephemeral timer system **202** incorporates a number of timers that, based on duration and display parameters associated with a message, or collection of messages (e.g., a story), selectively enable access (e.g., for presentation and display) to messages and associated content via the interaction application **104**. Further details regarding the operation of the ephemeral timer system **202** are provided below.

[0043] The collection management system **204** is responsible for managing sets or collections of media (e.g., collections of text, image video, and audio data). A collection of content (e.g., messages, including images, video, text, and audio) may be organized into an “event gallery” or an “event story.” Such a collection may be made available for a specified time period, such as the duration of an event to which the content relates. For example, content relating to a music concert may be made available as a “story” for the duration of that music concert. The collection management system **204** may also be responsible for publishing an icon that provides notification of the existence of a particular collection to the user interface of the interaction application **104**.

[0044] The collection management system **204** furthermore includes a curation interface **206** that allows a collection manager to manage and curate a particular collection of content. For example, the curation interface **206** enables an event organizer to curate a collection of content relating to a specific event (e.g., delete inappropriate content or redundant messages). Additionally, the collection management system **204** employs machine vision (or image recognition technology) and content rules to automatically curate a content collection. In certain examples, compensation may be paid to a user for the inclusion of user-generated content into a collection. In such cases, the

collection management system **204** operates to automatically make payments to such users for the use of their content.

[0045] The augmentation system **208** provides various functions that enable a user to augment (e.g., annotate or otherwise modify or edit) media content associated with a message. For example, the augmentation system **208** provides functions related to the generation and publishing of media overlays for messages processed by the system **100**. The augmentation system **208** operatively supplies a media overlay or augmentation (e.g., an image filter) to the interaction application **104** based on a geolocation of the user device **102**. In another example, the augmentation system **208** operatively supplies a media overlay to the interaction application **104** based on other information, such as social network information of the user of the user device **102**. A media overlay may include audio and visual content and visual effects. Examples of audio and visual content include pictures, texts, logos, animations, and sound effects. An example of a visual effect includes color overlaying. The audio and visual content or the visual effects can be applied to a media content item (e.g., a photo) at the user device **102**. For example, the media overlay may include text, a graphical element, or image that can be overlaid on top of a photograph taken by the user device **102**. In another example, the media overlay includes an identification of a location overlay (e.g., Venice beach), a name of a live event, or a name of a merchant overlay (e.g., Beach Coffee House). In another example, the augmentation system **208** uses the geolocation of the user device **102** to identify a media overlay that includes the name of a merchant at the geolocation of the user device **102**. The media overlay may include other indicia associated with the merchant. The media overlays may be stored in the database **126** and accessed through the database server **120**.

[0046] In some examples, the augmentation system **208** provides a user-based publication platform that enables users to select a geolocation on a map and upload content associated with the selected geolocation. The user may also specify circumstances under which a particular media overlay should be offered to other users. The augmentation system **208** generates a media overlay that includes the uploaded content and associates the uploaded content with the selected geolocation.

[0047] In other examples, the augmentation system **208** provides a merchant-based publication platform that enables merchants to select a particular media overlay associated with a geolocation via a bidding process. For example, the augmentation system **208** associates the media overlay of the highest bidding merchant with a corresponding geolocation for a predefined amount of time. The augmentation system **208** communicates with the image processing server **122** to obtain augmented reality experiences and presents identifiers of such experiences in one or more user interfaces (e.g., as icons over a real-time image or video or as thumbnails or icons in interfaces dedicated for presented identifiers of augmented reality experiences). Once an augmented reality experience is selected, one or more images, videos, or augmented reality graphical elements are retrieved and presented as an overlay on top of the images or video captured by the user device **102**. In some cases, the camera is switched to a front-facing view (e.g., the front-facing camera of the user device **102** is activated in response to activation of a particular augmented reality experience) and the images from the front-facing camera of the user device **102** start being displayed on the user device **102** instead of the rear-facing camera of the user device **102**. The one or more images, videos, or augmented reality graphical elements are retrieved and presented as an overlay on top of the images that are captured and displayed by the front-facing camera of the user device **102**.

[0048] The map system **210** provides various geographic location functions, and supports the presentation of map-based media content and messages by the interaction application **104**. For example, the map system **210** enables the display of user icons or avatars (e.g., stored in profile data **316**) on a map to indicate a current or past location of “friends” of a user, as well as media content (e.g., collections of messages including photographs and videos) generated by such friends, within the context of a map. For example, a message posted by a user to the system **100** from a specific geographic location may be displayed within the context of a map at that particular

location to “friends” of a specific user on a map interface of the interaction application **104**. A user can furthermore share his or her location and status information (e.g., using an appropriate status avatar) with other users of the system **100** via the interaction application **104**, with this location and status information being similarly displayed within the context of a map interface of the interaction application **104** to selected users.

[0049] The game system **212** provides various gaming functions within the context of the interaction application **104**. The interaction application **104** provides a game interface providing a list of available games (e.g., web-based games or web-based applications) that can be launched by a user within the context of the interaction application **104**, and played with other users of the system **100**. The system **100** further enables a particular user to invite other users to participate in the play of a specific game, by issuing invitations to such other users from the interaction application **104**. The interaction application **104** also supports both voice and text messaging (e.g., chats) within the context of gameplay, provides a leaderboard for the games, and also supports the provision of in-game rewards (e.g., coins and items).

[0050] The external resource system **220** provides an interface for the interaction application **104** to communicate with external app(s) servers **110** to launch or access external resources. Each external resource (apps) server **110** hosts, for example, a markup language (e.g., HTML5) based application or small-scale version of an external application (e.g., game, utility, payment, or ride-sharing application that is external to the interaction application **104**). The interaction application **104** may launch a web-based resource (e.g., application) by accessing the HTML5 file from the external resource (apps) servers **110** associated with the web-based resource. In certain examples, applications hosted by external resource servers **110** are programmed in JavaScript leveraging a Software Development Kit (SDK) provided by the interaction application server **118**. The SDK includes Application Programming Interfaces (APIs) with functions that can be called or invoked by the web-based application. In certain examples, the interaction application server **118** includes a JavaScript library that provides a given third-party resource access to certain user data of the interaction application **104**. HTML5 is used as an example technology for programming games, but applications and resources programmed based on other technologies can be used.

[0051] In order to integrate the functions of the SDK into the web-based resource, the SDK is downloaded by an external resource (apps) server **110** from the interaction application server **118** or is otherwise received by the external resource (apps) server **110**. Once downloaded or received, the SDK is included as part of the application code of a web-based external resource. The code of the web-based resource can then call or invoke certain functions of the SDK to integrate features of the interaction application **104** into the web-based resource.

[0052] The SDK stored on the interaction application server **118** effectively provides the bridge between an external resource (e.g., third-party or external applications **109** or applets and the interaction application **104**). This provides the user with a seamless experience of communicating with other users on the interaction application **104**, while also preserving the look and feel of the interaction application **104**. To bridge communications between an external resource and an interaction application **104**, in certain examples, the SDK facilitates communication between external resource servers **110** and the interaction application **104**. In certain examples, a Web ViewJavaScriptBridge running on a user device **102** establishes two one-way communication channels between an external resource and the interaction application **104**. Messages are sent between the external resource and the interaction application **104** via these communication channels asynchronously. Each SDK function invocation is sent as a message and callback. Each SDK function is implemented by constructing a unique callback identifier and sending a message with that callback identifier.

[0053] By using the SDK, not all information from the interaction application **104** is shared with external resource servers **110**. The SDK limits which information is shared based on the needs of the external resource. In certain examples, each external resource server **110** provides an HTML5

file corresponding to the web-based external resource to the interaction application server **118**. The interaction application server **118** can add a visual representation (such as a box art or other graphic) of the web-based external resource in the interaction application **104**. Once the user selects the visual representation or instructs the interaction application **104** through a GUI of the interaction application **104** to access features of the web-based external resource, the interaction application **104** obtains the HTML5 file and instantiates the resources necessary to access the features of the web-based external resource.

[0054] The interaction application **104** presents a graphical user interface (e.g., a landing page or title screen) for an external resource. During, before, or after presenting the landing page or title screen, the interaction application **104** determines whether the launched external resource has been previously authorized to access user data of the interaction application **104**. In response to determining that the launched external resource has been previously authorized to access user data of the interaction application **104**, the interaction application **104** presents another graphical user interface of the external resource that includes functions and features of the external resource. In response to determining that the launched external resource has not been previously authorized to access user data of the interaction application **104**, after a threshold period of time (e.g., 3 seconds) of displaying the landing page or title screen of the external resource, the interaction application **104** slides up (e.g., animates a menu as surfacing from a bottom of the screen to a middle of or other portion of the screen) a menu for authorizing the external resource to access the user data. The menu identifies the type of user data that the external resource will be authorized to use. In response to receiving a user selection of an accept option, the interaction application **104** adds the external resource to a list of authorized external resources and allows the external resource to access user data from the interaction application **104**. In some examples, the external resource is authorized by the interaction application **104** to access the user data in accordance with an OAuth 2 framework.

[0055] The interaction application **104** controls the type of user data that is shared with external resources based on the type of external resource being authorized. For example, external resources that include full-scale external applications (e.g., a third-party or external application **109**) are provided with access to a first type of user data (e.g., only two-dimensional avatars of users with or without different avatar characteristics). As another example, external resources that include small-scale versions of external applications (e.g., web-based versions of third-party applications) are provided with access to a second type of user data (e.g., payment information, two-dimensional avatars of users, three-dimensional avatars of users, and avatars with various avatar characteristics). Avatar characteristics include different ways to customize a look and feel of an avatar, such as different poses, facial features, clothing, and so forth.

Data Architecture

[0056] FIG. **3** is a schematic diagram illustrating data structures **300**, which may be stored in the database **126** of the server system **108**, according to certain examples. While the content of the database **126** is shown to comprise a number of tables, it will be appreciated that the data could be stored in other types of data structures (e.g., as an object-oriented database).

[0057] The database **126** includes message data stored within a message table **302**. This message data includes, for any particular one message, at least message sender data, message recipient (or receiver) data, and a payload. Further details regarding information that may be included in a message, and included within the message data stored in the message table **302**.

[0058] An entity table **306** stores entity data, and is linked (e.g., referentially) to an entity graph **308** and profile data **316**. Entities for which records are maintained within the entity table **306** may include individuals, corporate entities, organizations, objects, places, events, and so forth. Regardless of entity type, any entity regarding which the server system **108** stores data may be a recognized entity. Each entity is provided with a unique identifier, as well as an entity type identifier (not shown).

[0059] The entity graph **308** stores information regarding relationships and associations between entities. Such relationships may be social, professional (e.g., work at a common corporation or organization) interest-based or activity-based, merely for example.
[0060] The profile data **316** stores multiple types of profile data about a particular entity. The profile data **316** may be selectively used and presented to other users of the system **100**, based on privacy settings specified by a particular entity. Where the entity is an individual, the profile data **316** includes, for example, a user name, telephone number, address, settings (e.g., notification and privacy settings), as well as a user-selected avatar representation (or collection of such avatar representations). A particular user may then selectively include one or more of these avatar representations within the content of messages communicated via the system **100**, and on map interfaces displayed by interaction applications **104** to other users. The collection of avatar representations may include "status avatars," which present a graphical representation of a status or activity that the user may select to communicate at a particular time.
[0061] Where the entity is a group, the profile data **316** for the group may similarly include one or more avatar representations associated with the group, in addition to the group name, members, and various settings (e.g., notifications) for the relevant group.
[0062] The database **126** also stores augmentation data, such as overlays or filters, in an augmentation table **310**. The augmentation data is associated with and applied to videos (for which data is stored in a video table **304**) and images (for which data is stored in an image table **312**).
[0063] Filters, in one example, are overlays that are displayed as overlaid on an image or video during presentation to a recipient user. Filters may be of various types, including user-selected filters from a set of filters presented to a sending user by the interaction application **104** when the sending user is composing a message. Other types of filters include geolocation filters (also known as geo-filters), which may be presented to a sending user based on geographic location. For example, geolocation filters specific to a neighborhood or special location may be presented within a user interface by the interaction application **104**, based on geolocation information determined by a Global Positioning System (GPS) unit of the user device **102**.
[0064] Another type of filter is a data filter, which may be selectively presented to a sending user by the interaction application **104**, based on other inputs or information gathered by the user device **102** during the message creation process. Examples of data filters include current temperature at a specific location, a current speed at which a sending user is traveling, battery life for a user device **102**, or the current time.
[0065] Other augmentation data that may be stored within the image table **312** includes augmented reality content items (e.g., corresponding to applying augmented reality experiences). An augmented reality content item or augmented reality item may be a real-time special effect and sound that may be added to an image or a video.
[0066] As described above, augmentation data includes augmented reality content items (also referred to as virtual content items, virtual content, and/or virtual items), overlays, image transformations, AR images, and similar terms that refer to modifications that may be applied to image data (e.g., videos or images). This includes real-time modifications, which modify an image as it is captured using device sensors (e.g., one or multiple cameras) of a user device **102** and then displayed on a screen of the user device **102** with the modifications. This also includes modifications to stored content, such as video clips in a gallery that may be modified. For example, in a user device **102** with access to multiple augmented reality content items, a user can use a single video clip with multiple augmented reality content items to see how the different augmented reality content items will modify the stored clip. For example, multiple augmented reality content items that apply different pseudorandom movement models can be applied to the same content by selecting different augmented reality content items for the content. Similarly, real-time video capture may be used with an illustrated modification to show how video images currently being captured by sensors of a user device **102** would modify the captured data. Such data may simply be

displayed on the screen and not stored in memory, or the content captured by the device sensors may be recorded and stored in memory with or without the modifications (or both). In some systems, a preview feature can show how different augmented reality content items will look within different windows in a display at the same time. This can, for example, enable multiple windows with different pseudorandom animations to be viewed on a display at the same time.
[0067] Data and various systems using augmented reality content items or other such transform systems to modify content using this data can thus involve detection of objects (e.g., faces, hands, bodies, cats, dogs, surfaces, objects, etc.), tracking of such objects as they leave, enter, and move around the field of view in video frames, and the modification or transformation of such objects as they are tracked. In various examples, different methods for achieving such transformations may be used. Some examples may involve generating a three-dimensional mesh model of the object or objects, and using transformations and animated textures of the model within the video to achieve the transformation. In other examples, tracking of points on an object may be used to place an image or texture (which may be two dimensional or three dimensional) at the tracked position. In still further examples, neural network analysis of video frames may be used to place images, models, or textures in content (e.g., images or frames of video). Augmented reality content items thus refer both to the images, models, and textures used to create transformations in content, as well as to additional modeling and analysis information needed to achieve such transformations with object detection, tracking, and placement.
[0068] Real-time video processing can be performed with any kind of video data (e.g., video streams, video files, etc.) saved in a memory of a computerized system of any kind. For example, a user can load video files and save them in a memory of a device, or can generate a video stream using sensors of the device. Additionally, any objects can be processed using a computer animation model, such as a human's face and parts of a human body, animals, or non-living things such as chairs, cars, or other objects.
[0069] In some examples, when a particular modification is selected along with content to be transformed, elements to be transformed are identified by the computing device, and then detected and tracked if they are present in the frames of the video. The elements of the object are modified according to the request for modification, thus transforming the frames of the video stream. Transformation of frames of a video stream can be performed by different methods for different kinds of transformation. For example, for transformations of frames mostly referring to changing forms of object's elements, characteristic points for each element of an object are calculated (e.g., using an Active Shape Model (ASM) or other known methods). Then, a mesh based on the characteristic points is generated for each of the at least one element of the object. This mesh is used in the following stage of tracking the elements of the object in the video stream. In the process of tracking, the mentioned mesh for each element is aligned with a position of each element. Then, additional points are generated on the mesh. A first set of first points is generated for each element based on a request for modification, and a set of second points is generated for each element based on the set of first points and the request for modification. Then, the frames of the video stream can be transformed by modifying the elements of the object on the basis of the sets of first and second points and the mesh. In such method, a background of the modified object can be changed or distorted as well by tracking and modifying the background.
[0070] In some examples, transformations changing some areas of an object using its elements can be performed by calculating characteristic points for each element of an object and generating a mesh based on the calculated characteristic points. Points are generated on the mesh, and then various areas based on the points are generated. The elements of the object are then tracked by aligning the area for each element with a position for each of the at least one elements, and properties of the areas can be modified based on the request for modification, thus transforming the frames of the video stream. Depending on the specific request for modification, properties of the mentioned areas can be transformed in different ways. Such modifications may involve changing

color of areas; removing at least some part of areas from the frames of the video stream; including one or more new objects into areas which are based on a request for modification; and modifying or distorting the elements of an area or object. In various examples, any combination of such modifications or other similar modifications may be used. For certain models to be animated, some characteristic points can be selected as control points to be used in determining the entire state-space of options for the model animation.

[0071] In some examples of a computer animation model to transform image data using face detection, the face is detected on an image with use of a specific face detection algorithm (e.g., Viola-Jones). Then, an Active Shape Model (ASM) algorithm is applied to the face region of an image to detect facial feature reference points.

[0072] Other methods and algorithms suitable for face detection can be used. For example, in some examples, features are located using a landmark, which represents a distinguishable point present in most of the images under consideration. For facial landmarks, for example, the location of the left eye pupil may be used. If an initial landmark is not identifiable (e.g., if a person has an eyepatch), secondary landmarks may be used. Such landmark identification procedures may be used for any such objects. In some examples, a set of landmarks forms a shape. Shapes can be represented as vectors using the coordinates of the points in the shape. One shape is aligned to another with a similarity transform (allowing translation, scaling, and rotation) that minimizes the average Euclidean distance between shape points. The mean shape is the mean of the aligned training shapes.

[0073] In some examples, a search for landmarks from the mean shape aligned to the position and size of the face determined by a global face detector is started. Such a search then repeats the steps of suggesting a tentative shape by adjusting the locations of shape points by template matching of the image texture around each point and then conforming the tentative shape to a global shape model until convergence occurs. In some systems, individual template matches are unreliable, and the shape model pools the results of the weak template matches to form a stronger overall classifier. The entire search is repeated at each level in an image pyramid, from coarse to fine resolution.

[0074] A transformation system can capture an image or video stream on a client device (e.g., the user device **102**) and perform complex image manipulations locally on the user device **102** while maintaining a suitable user experience, computation time, and power consumption. The complex image manipulations may include size and shape changes, emotion transfers (e.g., changing a face from a frown to a smile), state transfers (e.g., aging a subject, reducing apparent age, changing gender), style transfers, graphical element application, and any other suitable image or video manipulation implemented by a convolutional neural network that has been configured to execute efficiently on the user device **102**.

[0075] In some examples, a computer animation model to transform image data can be used by a system where a user may capture an image or video stream of the user (e.g., a selfie) using a user device **102** having a neural network operating as part of the interaction application **104** operating on the user device **102**. The transformation system operating within the interaction application **104** determines the presence of a face within the image or video stream and provides modification icons associated with a computer animation model to transform image data, or the computer animation model can be present as associated with an interface described herein. The modification icons include changes that may be the basis for modifying the user's face within the image or video stream as part of the modification operation. Once a modification icon is selected, the transformation system initiates a process to convert the image of the user to reflect the selected modification icon (e.g., generate a smiling face on the user). A modified image or video stream may be presented in a graphical user interface displayed on the user device **102** as soon as the image or video stream is captured, and a specified modification is selected. The transformation system may implement a complex convolutional neural network on a portion of the image or video stream to generate and apply the selected modification. That is, the user may capture the image or video

stream and be presented with a modified result in real-time or near real-time once a modification icon has been selected. Further, the modification may be persistent while the video stream is being captured, and the selected modification icon remains toggled. Machine-taught neural networks may be used to enable such modifications.

[0076] The graphical user interface, presenting the modification performed by the transformation system, may supply the user with additional interaction options. Such options may be based on the interface used to initiate the content capture and selection of a particular computer animation model (e.g., initiation from a content creator user interface). In various examples, a modification may be persistent after an initial selection of a modification icon. The user may toggle the modification on or off by tapping or otherwise selecting the face being modified by the transformation system and store it for later viewing or browse to other areas of the imaging application. Where multiple faces are modified by the transformation system, the user may toggle the modification on or off globally by tapping or selecting a single face modified and displayed within a graphical user interface. In some examples, individual faces, among a group of multiple faces, may be individually modified, or such modifications may be individually toggled by tapping or selecting the individual face or a series of individual faces displayed within the graphical user interface.

[0077] A story table **314** stores data regarding collections of messages and associated images, video, or audio data, which are compiled into a collection (e.g., a story or a gallery). The creation of a particular collection may be initiated by a particular user (e.g., each user for which a record is maintained in the entity table **306**). A user may create a “personal story” in the form of a collection of content that has been created and sent/broadcast by that user. To this end, the user interface of the interaction application **104** may include an icon that is user-selectable to enable a sending user to add specific content to his or her personal story.

[0078] A collection may also constitute a “live story,” which is a collection of content from multiple users that is created manually, automatically, or using a combination of manual and automatic techniques. For example, a “live story” may constitute a curated stream of user-submitted content from various locations and events. Users whose client devices have location services enabled and are at a common location event at a particular time may, for example, be presented with an option, via a user interface of the interaction application **104**, to contribute content to a particular live story. The live story may be identified to the user by the interaction application **104**, based on his or her location. The end result is a “live story” told from a community perspective.

[0079] A further type of content collection is known as a “location story,” which enables a user whose user device **102** is located within a specific geographic location (e.g., on a college or university campus) to contribute to a particular collection. In some examples, a contribution to a location story may require a second degree of authentication to verify that the end user belongs to a specific organization or other entity (e.g., is a student on the university campus).

[0080] As mentioned above, the video table **304** stores video data that, in one example, is associated with messages for which records are maintained within the message table **302**. Similarly, the image table **312** stores image data associated with messages for which message data is stored in the entity table **306**. The entity table **306** may associate various augmentations from the augmentation table **310** with various images and videos stored in the image table **312** and the video table **304**.

Eyewear Device

[0081] FIG. **4** shows a front perspective view of an eyewear device **119** in the form of a pair of smart glasses that include a content centering system **107** according to one example. The eyewear device **119** includes a body **403** including a front piece or frame **406** and a pair of temples **409** connected to the frame **406** for supporting the frame **406** in position on a user's face **437** when the eyewear device **119** is worn. The frame **406** can be made from any suitable material such as plastics or metal, including any suitable shape memory alloy. The eyewear device **119** can include a button

402 for capturing a content item (image or video).
[0082] The eyewear device **119** includes a pair of optical elements in the form of a pair of lenses **412** held by corresponding optical element holders in the form of a pair of rims **415** forming part of the frame **406**. The rims **415** are connected by a bridge **418**. In other examples, one or both of the optical elements can be a display, a display assembly, or a lens and display combination.
[0083] The frame **406** includes a pair of end pieces **421** defining lateral end portions of the frame **406**. In this example, a variety of electronics components are housed in one or both of the end pieces **421**. The temples **409** are coupled to the respective end pieces **421**. In this example, the temples **409** are coupled to the frame **406** by respective hinges so as to be hingedly movable between a wearable mode and a collapsed mode in which the temples **409** are pivoted towards the frame **406** to lie substantially flat against it. In other examples, the temples **409** can be coupled to the frame **406** by any suitable means, or can be rigidly or fixedly secured to the frame **406** so as to be integral therewith.
[0084] Each of the temples **409** that includes a front portion of that is coupled to the frame **406** and any suitable rear portion for coupling to the ear of the user, such as the curves or cute piece illustrated in the example of FIG. **4**. In some examples, the frame **406** is formed of a single piece of material, so as to have a unitary or monolithic construction. In some examples, the whole of the body **403** (including both the frame **406** and the temples **409**) can be of the unitary or monolithic construction.
[0085] The eyewear device **119** has onboard electronics components including a computing device, such as a computer **424**, or low power processor, which can in different examples be of any suitable type so as to be carried by the body **403**. In some examples, the computer **424** is at least partially housed in one or both of the temples **409**. In the present example, various components of the computer **424** are housed in the lateral end pieces **421** of the frame **406**. The computer **424** includes one or more processors with memory (e.g., a volatile storage device, such as random access memory or registers), a storage device (e.g., a non-volatile storage device), wireless communication circuitry (e.g., BLE communication devices and/or WiFi direct devices), and a power source. The computer **424** comprises low-power circuitry, high-speed circuitry, and, in some examples, a display processor. Various examples may include these elements in different configurations or integrated together in different ways.
[0086] The computer **424** additionally includes a battery **427** or other suitable portable power supply. In one example, the battery **427** is disposed in one of the temples **409**. In the eyewear device **119** shown in FIG. **4**, the battery **427** is shown as being disposed in one of the end pieces **421**, being electrically coupled to the remainder of the computer **424** housed in the corresponding end piece **421**.
[0087] The eyewear device **119** is camera-enabled, in this example including a camera **430** mounted in one of the end pieces **421** and facing forwards so as to be aligned more or less with the direction of view of a wearer of the eyewear device **119**. The camera **430** is configured to capture digital images (also referred to herein as digital photographs or pictures) as well as digital video content. Operation of the camera **430** is controlled by a camera controller provided by the computer **424**, image data representative of images or video captured by the camera **430** being temporarily stored on a memory forming part of the computer **424**. In some examples, the eyewear device **119** can have a pair of cameras **430**, e.g. housed by the respective end pieces **421**.
[0088] The eyewear device **119** (e.g., the AR glasses or VR devices) has computer vision capabilities and cameras that help orient virtual content in 6D of real world positions. The camera capabilities of the eyewear device **119** track the user's hands in real time to enable the user to control the virtual content using hand gestures. The camera **430** of the eyewear device **119** can include multiple video cameras that communicate with each other and with the on-board computer **424** to position the AR/VR content and track a user's hand position relative to the AR/VR content to control interactions with the AR/VR content.

[0089] As will be described in greater detail below, the onboard computer **424** and the lenses **412** are configured together to provide a content centering system **107** that automatically and selectively re-centers virtual content to bring the virtual content to within view of the lenses **412** by moving the virtual content from a first virtual location to a second virtual location. Specifically, the lenses **412** can display virtual content or one or more virtual objects. This makes it appear to the user that the virtual content is integrated within a real-world environment that the user views through the lenses **412**. In some examples, the virtual content is received from the user device **102**. In some examples, the virtual content is received directly from the application servers **114**. The eyewear device **119** uses "computer vision processing" to position virtual content in the user's real world and tracks the position of the user's hands for real time input to the system.
[0090] The eyewear device **119** may include an accelerometer and a touch interface and a voice command system. Based on input received by the eyewear device **119** from the accelerometer and a touch interface and the voice command system, the eyewear device **119** can control user interaction with the virtual content. In some cases, the eyewear device **119**, rather than using a touch interface to control virtual content interactions (or in addition to using the touch interface) uses front-facing cameras that can track the user's hands in real time via computer vision technologies. The eyewear device **119** can then control interactions with the virtual content based on positioning of the user's hands and gestures performed by the hands. In one example, the user interaction can control playback of content that is presented on the lenses **412**. In another example, the user interaction can navigate through a playlist or music or video library. In another example, the user interaction can navigate through a conversation the user is involved in, such as by scrolling through various three-dimensional or two-dimensional conversation elements (e.g., chat bubbles) and selecting individual conversation elements to respond to generate messages to transmit to participants of the conversation.
[0091] The content centering system **107** (which can be implemented by the computer **424**) assigns virtual content to virtual locations. The content centering system **107** monitors the current virtual location that is within view of a real-world environment. The content centering system **107** retrieves virtual content for display that is within a specified range of the current virtual location that is within view and at a specified virtual offset which may be set or adjusted by the user. As the eyewear device **119** is moved around to be directed to a new portion of the real-world environment, associated with a different set of virtual locations, the content centering system **107** updates the virtual location of the virtual content to maintain or keep the virtual content displayed at substantially the same virtual offset relative to a point (e.g., lenses) on the eyewear device **119**. In some cases, the user is controlling and interacting with the virtual content using hand gestures. In these circumstances, it becomes important to position the virtual objects at a comfortable place in 3D space relative to the user. The disclosed techniques allow the user to control that specific position that is comfortable for the user to interact with the virtual content using the user's hands.
[0092] The eyewear device **119** further includes one or more communication devices, such as Bluetooth low energy (BLE) communication interface. Such BLE communication interface enables the eyewear device **119** to communicate wirelessly with the user device **102**. Other forms of wireless communication can also be employed instead of, or in addition to, the BLE communication interface, such as a WiFi direct interface. The BLE communication interface implements a standard number of BLE communication protocols.
[0093] A first of the communications protocols implemented by the BLE interface of the eyewear device **119** enables an unencrypted link to be established between the eyewear device **119** and the user device **102**. In this first protocol, the link-layer communication (the physical interface or medium) between the eyewear device **119** and the user device **102** includes unencrypted data. In this first protocol, the application layer (the communication layer operating on the physically exchanged data) encrypts and decrypts data that is physically exchanged in unencrypted form over the link layer of the BLE communication interface. In this way, data exchanged over the physical

layer can freely be read by an eavesdropping device, but the eavesdropping device will not be able to decipher the data that is exchanged without performing a decryption operation in the application layer.

[0094] A second of the communications protocols implemented by the BLE interface of the eyewear device **119** enables an encrypted link to be established between the eyewear device **119** and the user device **102**. In this second protocol, the link-layer communication (the physical interface) between the eyewear device **119** and the user device **102** receives data from the application layer and adds a first type of encryption to the data before exchanging the data over the physical medium. In this second protocol, the application layer (the communication layer operating on the physically exchanged data) may or may not use a second type of encryption to encrypt and decrypt data that is physically exchanged in encrypted form, using the first type of encryption, over the link layer of the BLE communication interface. Namely, data can be first encrypted by the application layer and then be further encrypted by the physical layer before being exchanged over the physical medium. Following the exchange over the physical medium, the data is then decrypted by the physical layer and then decrypted again (e.g., using a different type of encryption) by the application layer. In this way, data exchanged over the physical layer cannot be read by an eavesdropping device as the data is encrypted in the physical medium.

[0095] In some examples, the user device **102** communicates with the eyewear device **119** using the first protocol to exchange images or videos or virtual content between the interaction application **104** and the eyewear device **119**.

Content Centering System

[0096] FIG. **5** is a flowchart illustrating example operations of the content centering system **107** in performing a process or method **500**, according to examples. The process or method **500** may be embodied in computer-readable instructions for execution by one or more processors such that the operations of the process or method **500** may be performed in part or in whole by the functional components of the notification management system **107**; accordingly, the process or method **500** is described below by way of example with reference thereto. However, in other examples, at least some of the operations of the process or method **500** may be deployed on various other hardware configurations. The process or method **500** is therefore not intended to be limited to the content centering system **107** and can be implemented in whole, or in part, by any other component. Some or all of the operations of process or method **500** can be in parallel, out of order, or entirely omitted.

[0097] At operation **501**, the content centering system **107** displays, on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space, as discussed above and below.

[0098] At operation **502**, the content centering system **107** receives a request to move the one or more virtual objects to a second virtual coordinate in 3D space, as discussed above and below. For example, one or more cameras of the content centering system **107** can identify one or more hands of the user at a position in 3D space corresponding to the first virtual coordinate. In response, the content centering system **107** can determine that the one or more hands have performed a given gesture associated with dragging or moving a virtual object. In such cases, the content centering system **107** tracks movement of the hands in 3D space to identify the second virtual coordinate to which the hands have been moved. The content centering system **107** determines that the hands have performed another gesture associated with releasing the virtual object. In such cases, the content centering system **107** updates the virtual position of the virtual objects from being displayed at the first virtual coordinate to the second virtual coordinate.

[0099] At operation **503**, the content centering system **107** in response to receiving the request, maintains display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device, as discussed above and below. Specifically, if a user additionally moves the virtual content using a hand

interaction or other input method, independent of their body motion, the virtual content remains visible within the user's display FOV. The virtual content is prevented from being moved beyond/outside of the user's display FOV. If the user tries to move content outside of the FOV, the content centering system **107** may gently nudge (move or animate) the virtual content back into the visible FOV, still following the overall position of the AR device.

[0100] FIGS. **6-8** are illustrative screens of a graphical user interface of the content centering system **107**, according to some examples. The screens shown in FIGS. **6-8** may be provided by the interaction application **104** of one or more client devices **102**, other applications implemented on one or more client devices **102**, and/or the eyewear device **119**.

[0101] For example, the screen **600** shows a view of a first portion of a real-world environment **601** through the lenses **412** of the eyewear device **119**. The eyewear device **119** may receive input from the user that requests to view a list of thumbnails representing a music or video library (or other virtual content item). In response, the eyewear device **119** obtains from the user device **102** and/or from the application servers **114** one or more virtual objects **602**. The virtual objects **602** include thumbnails (e.g., cover art) representing different media assets (e.g., different songs or videos). The eyewear device **119** obtains default virtual display positions or virtual coordinates at which to display the virtual objects **602** from the user device **102** and/or the application servers **114**.

[0102] The eyewear device **119** associates or places the virtual objects **602** at a first virtual location that is associated with the first portion of the real-world environment **601** based on the default virtual display position received by the eyewear device **119**. In some cases, the eyewear device **119** displays the virtual objects **602** as 3D objects in a center of the lenses **412**, at a default virtual height relative to a real-world surface visible through the lenses of the eyewear device **119** and at a default virtual distance relative to a point of the eyewear device **119**. This makes it appear to the user viewing the first portion of the real-world environment **601** through the lenses **412** as if the virtual objects **602** are within the first portion of the real-world environment **601** at the first virtual location.

[0103] The eyewear device **119** can receive input from the user (e.g., via computer-vision recognized hand gestures/poses, verbally and/or via touch input or from the user device **102**) to interact with the virtual objects **602**. For example, the user can scroll through the virtual objects **602**. The virtual objects **602** that are lower in the list can appear to be further away from the user than those that are earlier in the list. As the user scrolls through the virtual objects **602**, the virtual objects that are later in the list, are enlarged and are brought into view closer to the user. The user can provide an input to select a given one of the virtual objects **602**. In response, the media asset associated with the virtual object that is selected is retrieved and presented to the user. The user can pause and fast-forward and rewind playback of the media asset or perform any other function represented by a menu displayed as the virtual objects **602**.

[0104] In some cases, the user moves around while wearing the eyewear device **119**. For example, the user can turn their head to face another direction which results in a second portion of the real-world environment being viewed through the lenses **412** of the eyewear device **119**. As the user turns their head, the virtual display position of the virtual objects **602** is updated to maintain display of the virtual objects **602** at the virtual offset relative to the user and/or point on the eyewear device **119**. For example, if the virtual objects **602** were presented at a particular virtual height and virtual distance relative to the eyewear device **119** when the first portion of the real-world environment was being viewed, the virtual objects **602** can be presented at the same particular virtual height in the real world and virtual distance relative to the eyewear device **119** when the second portion of the real-world environment is being viewed.

[0105] In some cases, the eyewear device **119** receives input that updates the virtual offset associated with the virtual objects **602**. For example, the input can request to modify the virtual height and/or the virtual distance relative to the point on the eyewear device **119** at which the virtual objects **602** are being displayed. This input can be received verbally or by the user selecting

and dragging the virtual objects **602** to a desired virtual position in 3D space. In response, the eyewear device **119** updates the default virtual offset associated with the virtual objects **602**. For example, as shown in FIG. **7**, the screen **700** shows the virtual objects **702** (corresponding to the virtual objects **602**) at a new virtual offset displayed on the real-world environment **701**. Namely, the virtual objects **702** are presented closer to the point on the eyewear device **119** and lower in height than the virtual objects **602** presented in FIG. **6**.

[0106] In some cases, the eyewear device **119** computes a new virtual offset by measuring the virtual distance between the point on the eyewear device **119** and the virtual objects **602** and by measuring the virtual height of the virtual objects **602** relative to a point on a surface of the real-world environment visible through the lenses of the eyewear device **119**. The eyewear device **119** stores this new virtual offset in association with the virtual objects **602** so that when the virtual objects **602** is presented again at a future time (after being removed from being displayed), the virtual objects **602** is presented at the same virtual offset as that stored in association with the virtual objects **602**.

[0107] In some cases, the eyewear device **119** can maintain presentation of the virtual objects **602** at the virtual offset as the eyewear device **119** is moved around to view other portions of the real-world environment. In some cases, the eyewear device **119** determines a type associated with the virtual objects **602**. The eyewear device **119** updates a database that associates types of virtual objects with different virtual offsets based on the new virtual offset that is computed. For example, the eyewear device **119** can store or access a database that associates virtual thumbnails for music items with a first virtual offset and virtual menu items for AR experiences with a second virtual offset. When the user requests to access the menu for the virtual thumbnails for the music items, the eyewear device **119** presents the virtual thumbnails at the first virtual offset. In response to a request to access the menu items for the AR experiences, the eyewear device **119** presents the menu for the AR experiences at the second virtual offset. The eyewear device **119** can determine that the virtual objects **602** corresponds to virtual thumbnails for music items. In response, the eyewear device **119** updates the first virtual offset stored in the database to be the newly computed virtual offset and does not update the second virtual offset because the eyewear device **119** determines that the first virtual offset is associated with the same type of virtual object as the virtual objects **602**.

[0108] In some cases, the eyewear device **119** receives input that drags the virtual objects **702** outside of the FOV of the user of the eyewear device **119** and/or one or more cameras of the eyewear device **119**. The eyewear device **119** can receive input (via computer-vision recognized hand gestures/poses, verbally and/or using a cursor) that selects (e.g., taps) on the virtual objects **702** or places their hands over a position of the virtual objects **702**. The input can then drag or move the virtual objects **702** outside of the FOV so that it is no longer visible in the real-world environment **701** that is perceived through the lenses of the eyewear device **119**. At this point, the eyewear device **119** receives input that releases the virtual objects **702** while the virtual objects **702** are outside the FOV. In response to receiving the input that places and releases the virtual objects **702** outside of the FOV of the eyewear device **119**, the eyewear device **119** automatically repositions the virtual objects **702** at a virtual coordinate that is within the FOV of the eyewear device **119** or the user. The virtual coordinate can correspond to the virtual offset or some other user specified point in 3D space.

[0109] In some examples, the eyewear device **119** receives a user request to access another set or type of virtual object, such as the virtual object **801** shown in the user interface **800** of FIG. **8**. The eyewear device **119** retrieves the default virtual coordinate associated with the virtual object **801**. The eyewear device **119** determines whether the type of the virtual object **801** matches one of the types of virtual objects for which the user previously defined a virtual offset. If so, the eyewear device **119** automatically repositions and adjusts the default virtual coordinate of the virtual object **801** so that the virtual object **801** is positioned in 3D space at the same or substantially the same virtual coordinate or virtual offset as the virtual objects **702** that was previously displayed by the

eyewear device **119**.

[0110] In response to the eyewear device **119** determining that the virtual object **801** is of a type that is not associated with a virtual offset previously defined by the user, the eyewear device **119** can present the virtual object **801** at the default virtual coordinate. The eyewear device **119** can present a prompt or message for the user of the eyewear device **119** informing the user that the default position can be adjusted for the virtual object **801**. The eyewear device **119** can receive input that moves the virtual object **801** to a new virtual coordinate. The eyewear device **119** computes the virtual offset based on the new virtual coordinate relative to the user and associates that virtual offset with the type of virtual content corresponding to the virtual object **801**. The eyewear device **119** can present a prompt requesting the user to confirm whether to associate the virtual offset of the virtual object **801** with all other virtual content that is of the same type as the virtual object **801**. In response to receiving input from the user confirming this operation, the eyewear device **119** automatically updates the default virtual coordinate for that type of virtual content with the virtual offset of the virtual object **801**.

Examples

[0111] Example 1. A method comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.

[0112] Example 2. The method of Example 1, wherein the first virtual coordinate comprises a default position for the one or more virtual objects.

[0113] Example 3. The method of Example 2, further comprising: determining a type associated with the one or more virtual objects; and obtaining the default position based on the type associated with the one or more virtual objects.

[0114] Example 4. The method of Example 3, further comprising: associating the second virtual coordinate with the type associated with the one or more virtual objects to replace the default position with the second virtual coordinate.

[0115] Example 5. The method of any one of Examples 1-4, further comprising: computing a virtual offset between the second virtual coordinate of the one or more virtual objects and a point on the user device, the virtual offset defining a 3D height relative to a surface of the real-world environment and a virtual distance between the point and the second virtual coordinate; and storing the virtual offset in a profile associated with a user.

[0116] Example 6. The method of Example 5, further comprising: receiving a request to access an additional virtual object; retrieving a default virtual position of the additional virtual object; computing a target virtual position by adjusting the default virtual position based on the virtual offset; and displaying the additional virtual object at the target virtual position.

[0117] Example 7. The method of Example 6, further comprising: displaying the additional virtual object at a same 3D height relative to the surface of the real-world environment and a same virtual distance between the point of the user device and the second virtual coordinate as the 3D height and the virtual distance of the one or more virtual objects.

[0118] Example 8. The method of any one of Examples 1-7, further comprising: receiving input that moves the one or more virtual objects to a third virtual coordinate in 3D space that is outside of a field of view (FOV) of the user device.

[0119] Example 9. The method of Example 8, further comprising: in response to receiving the input that moves the one or more virtual objects to a third virtual coordinate in 3D space that is outside of the FOV of the user device, automatically positioning the one or more virtual objects at a predetermined virtual coordinate in 3D space, the predetermined virtual coordinate corresponding to a position that is inside of the FOV of the user device.

[0120] Example 10. The method of Example 9, further comprising: receiving input that defines the predetermined virtual coordinate prior to receiving the input that moves the one or more virtual objects to the third virtual coordinate.
[0121] Example 11. The method of any one of Examples 9-10, wherein the one or more virtual objects are automatically positioned at the predetermined virtual coordinate in 3D space in response to receiving a command to terminate moving of the one or more virtual objects to the third virtual coordinate in 3D space that is outside of the field of view (FOV) of the user device.
[0122] Example 12. The method of any one of Examples 1-11, wherein the user device comprises an augmented reality (AR) eyewear device.
[0123] Example 13. The method of any one of Examples 1-12, wherein the one or more objects comprise one or more thumbnails representing respective media assets.
[0124] Example 14. The method of any one of Examples 1-13, wherein the one or more objects comprise conversation elements that represent a conversation in which a user of the user device is involved.
[0125] Example 15. The method of any one of Examples 1-14, wherein the one or more objects comprise user interface elements for controlling and playing back one or more content items.
[0126] Example 16. The method of any one of Examples 1-15, wherein the one or more objects comprise user interface elements for generating augmented reality content.
[0127] Example 17. The method of claim **1**, wherein receiving the request comprises: detecting by one or more cameras of the user device a position of hands of the user at the first virtual coordinate; determining that the hands of the user perform a gesture associating with moving the one or more virtual objects; and determining that the hands have been moved to the second virtual coordinate.
[0128] Example 18. A system comprising: a storage device; and at least one processor configured to perform operations comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.
[0129] Example 19. The system of Example 18, the operations further comprising: determining a type associated with the one or more virtual objects; and obtaining the default position based on the type associated with the one or more virtual objects.
[0130] Example 20. A non-transitory machine-readable storage medium comprising instructions that, when executed by one or more processors of a machine, cause the machine to perform operations comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.

Machine Architecture

[0131] FIG. **9** is a diagrammatic representation of the machine **900** within which instructions **908** (e.g., software, a program, an application, an applet, an app, or other executable code) for causing the machine **900** to perform any one or more of the methodologies discussed herein may be executed. For example, the instructions **908** may cause the machine **900** to execute any one or more of the methods described herein. The instructions **908** transform the general, non-programmed machine **900** into a particular machine **900** programmed to carry out the described and illustrated functions in the manner described. The machine **900** may operate as a standalone device or may be coupled (e.g., networked) to other machines. In a networked deployment, the machine **900** may operate in the capacity of a server machine or a client machine in a server-client network

environment, or as a peer machine in a peer-to-peer (or distributed) network environment. The machine **900** may comprise, but not be limited to, a server computer, a client computer, a personal computer (PC), a tablet computer, a laptop computer, a netbook, a set-top box (STB), a personal digital assistant (PDA), an entertainment media system, a cellular telephone, a smartphone, a mobile device, a wearable device (e.g., a smartwatch), a smart home device (e.g., a smart appliance), other smart devices, a web appliance, a network router, a network switch, a network bridge, or any machine capable of executing the instructions **908**, sequentially or otherwise, that specify actions to be taken by the machine **900**. Further, while only a single machine **900** is illustrated, the term "machine" shall also be taken to include a collection of machines that individually or jointly execute the instructions **908** to perform any one or more of the methodologies discussed herein. The machine **900**, for example, may comprise the user device **102** or any one of a number of server devices forming part of the server system **108**. In some examples, the machine **900** may also comprise both client and server systems, with certain operations of a particular method or algorithm being performed on the server-side and with certain operations of the particular method or algorithm being performed on the client-side.

[0132] The machine **900** may include processors **902**, memory **904**, and input/output (I/O) components **938**, which may be configured to communicate with each other via a bus **940**. In an example, the processors **902** (e.g., a Central Processing Unit (CPU), a Reduced Instruction Set Computing (RISC) Processor, a Complex Instruction Set Computing (CISC) Processor, a Graphics Processing Unit (GPU), a Digital Signal Processor (DSP), an Application Specific Integrated Circuit (ASIC), a Radio-Frequency Integrated Circuit (RFIC), another processor, or any suitable combination thereof) may include, for example, a processor **906** and a processor **910** that execute the instructions **908**. The term "processor" is intended to include multi-core processors that may comprise two or more independent processors (sometimes referred to as "cores") that may execute instructions contemporaneously. Although FIG. **9** shows multiple processors **902**, the machine **900** may include a single processor with a single-core, a single processor with multiple cores (e.g., a multi-core processor), multiple processors with a single core, multiple processors with multiples cores, or any combination thereof.

[0133] The memory **904** includes a main memory **912**, a static memory **914**, and a storage unit **916**, all accessible to the processors **902** via the bus **940**. The main memory **904**, the static memory **914**, and the storage unit **916** store the instructions **908** embodying any one or more of the methodologies or functions described herein. The instructions **908** may also reside, completely or partially, within the main memory **912**, within the static memory **914**, within machine-readable medium **918** within the storage unit **916**, within at least one of the processors **902** (e.g., within the processor's cache memory), or any suitable combination thereof, during execution thereof by the machine **900**.

[0134] The I/O components **938** may include a wide variety of components to receive input, provide output, produce output, transmit information, exchange information, capture measurements, and so on. The specific I/O components **938** that are included in a particular machine will depend on the type of machine. For example, portable machines such as mobile phones may include a touch input device or other such input mechanisms, while a headless server machine will likely not include such a touch input device. It will be appreciated that the I/O components **938** may include many other components that are not shown in FIG. **9**. In various examples, the I/O components **938** may include user output components **924** and user input components **926**. The user output components **924** may include visual components (e.g., a display such as a plasma display panel (PDP), a light-emitting diode (LED) display, a liquid crystal display (LCD), a projector, or a cathode ray tube (CRT)), acoustic components (e.g., speakers), haptic components (e.g., a vibratory motor, resistance mechanisms), other signal generators, and so forth. The user input components **926** may include alphanumeric input components (e.g., a keyboard, a touch screen configured to receive alphanumeric input, a photo-optical keyboard, or other

alphanumeric input components), point-based input components (e.g., a mouse, a touchpad, a trackball, a joystick, a motion sensor, or another pointing instrument), tactile input components (e.g., a physical button, a touch screen that provides location and force of touches or touch gestures, or other tactile input components), audio input components (e.g., a microphone), and the like.

[0135] In further examples, the I/O components **938** may include biometric components **928**, motion components **930**, environmental components **932**, or position components **934**, among a wide array of other components. For example, the biometric components **928** include components to detect expressions (e.g., hand expressions, facial expressions, vocal expressions, body gestures, or eye-tracking), measure biosignals (e.g., blood pressure, heart rate, body temperature, perspiration, or brain waves), identify a person (e.g., voice identification, retinal identification, facial identification, fingerprint identification, or electroencephalogram-based identification), and the like. The motion components **930** include acceleration sensor components (e.g., accelerometer), gravitation sensor components, and rotation sensor components (e.g., gyroscope).

[0136] The environmental components **932** include, for example, one or cameras (with still image/photograph and video capabilities), illumination sensor components (e.g., photometer), temperature sensor components (e.g., one or more thermometers that detect ambient temperature), humidity sensor components, pressure sensor components (e.g., barometer), acoustic sensor components (e.g., one or more microphones that detect background noise), proximity sensor components (e.g., infrared sensors that detect nearby objects), gas sensors (e.g., gas detection sensors to detection concentrations of hazardous gases for safety or to measure pollutants in the atmosphere), or other components that may provide indications, measurements, or signals corresponding to a surrounding physical environment.

[0137] With respect to cameras, the user device **102** may have a camera system comprising, for example, front cameras on a front surface of the user device **102** and rear cameras on a rear surface of the user device **102**. The front cameras may, for example, be used to capture still images and video of a user of the user device **102** (e.g., “selfies”), which may then be augmented with augmentation data (e.g., filters) described above. The rear cameras may, for example, be used to capture still images and videos in a more traditional camera mode, with these images similarly being augmented with augmentation data. In addition to front and rear cameras, the user device **102** may also include a 360° camera for capturing 360° photographs and videos.

[0138] Further, the camera system of a user device **102** may include dual rear cameras (e.g., a primary camera as well as a depth-sensing camera), or even triple, quad or penta rear camera configurations on the front and rear sides of the user device **102**. These multiple cameras systems may include a wide camera, an ultra-wide camera, a telephoto camera, a macro camera, and a depth sensor, for example.

[0139] The position components **934** include location sensor components (e.g., a GPS receiver component), altitude sensor components (e.g., altimeters or barometers that detect air pressure from which altitude may be derived), orientation sensor components (e.g., magnetometers), and the like.

[0140] Communication may be implemented using a wide variety of technologies. The I/O components **938** further include communication components **936** operable to couple the machine **900** to a network **920** or devices **922** via respective coupling or connections. For example, the communication components **936** may include a network interface component or another suitable device to interface with the network **920**. In further examples, the communication components **936** may include wired communication components, wireless communication components, cellular communication components, Near Field Communication (NFC) components, Bluetooth® components (e.g., Bluetooth® Low Energy), Wi-Fi® components, and other communication components to provide communication via other modalities. The devices **922** may be another machine or any of a wide variety of peripheral devices (e.g., a peripheral device coupled via a USB).

[0141] Moreover, the communication components **936** may detect identifiers or include components operable to detect identifiers. For example, the communication components **936** may include Radio Frequency Identification (RFID) tag reader components, NFC smart tag detection components, optical reader components (e.g., an optical sensor to detect one-dimensional bar codes such as Universal Product Code (UPC) bar code, multi-dimensional bar codes such as Quick Response (QR) code, Aztec code, Data Matrix, Dataglyph, MaxiCode, PDF417, Ultra Code, UCC RSS-2D bar code, and other optical codes), or acoustic detection components (e.g., microphones to identify tagged audio signals). In addition, a variety of information may be derived via the communication components **936**, such as location via Internet Protocol (IP) geolocation, location via Wi-Fi® signal triangulation, location via detecting an NFC beacon signal that may indicate a particular location, and so forth.

[0142] The various memories (e.g., main memory **912**, static memory **914**, and memory of the processors **902**) and storage unit **916** may store one or more sets of instructions and data structures (e.g., software) embodying or used by any one or more of the methodologies or functions described herein. These instructions (e.g., the instructions **908**), when executed by processors **902**, cause various operations to implement the disclosed examples.

[0143] The instructions **908** may be transmitted or received over the network **920**, using a transmission medium, via a network interface device (e.g., a network interface component included in the communication components **936**) and using any one of several well-known transfer protocols (e.g., hypertext transfer protocol (HTTP). Similarly, the instructions **908** may be transmitted or received using a transmission medium via a coupling (e.g., a peer-to-peer coupling) to the devices **922**.

Software Architecture

[0144] FIG. **10** is a block diagram **1000** illustrating a software architecture **1004**, which can be installed on any one or more of the devices described herein. The software architecture **1004** is supported by hardware such as a machine **1002** that includes processors **1020**, memory **1026**, and I/O components **1038**. In this example, the software architecture **1004** can be conceptualized as a stack of layers, where each layer provides a particular functionality. The software architecture **1004** includes layers such as an operating system **1012**, libraries **1010**, frameworks **1008**, and applications **1006**. Operationally, the applications **1006** invoke API calls **1050** through the software stack and receive messages **1052** in response to the API calls **1050**.

[0145] The operating system **1012** manages hardware resources and provides common services. The operating system **1012** includes, for example, a kernel **1014**, services **1016**, and drivers **1022**. The kernel **1014** acts as an abstraction layer between the hardware and the other software layers. For example, the kernel **1014** provides memory management, processor management (e.g., scheduling), component management, networking, and security settings, among other functionality. The services **1016** can provide other common services for the other software layers. The drivers **1022** are responsible for controlling or interfacing with the underlying hardware. For instance, the drivers **1022** can include display drivers, camera drivers, BLUETOOTH® or BLUETOOTH® Low Energy drivers, flash memory drivers, serial communication drivers (e.g., USB drivers), WI-FI® drivers, audio drivers, power management drivers, and so forth.

[0146] The libraries **1010** provide a common low-level infrastructure used by the applications **1006**. The libraries **1010** can include system libraries **1018** (e.g., C standard library) that provide functions such as memory allocation functions, string manipulation functions, mathematic functions, and the like. In addition, the libraries **1010** can include API libraries **1024** such as media libraries (e.g., libraries to support presentation and manipulation of various media formats such as Moving Picture Experts Group-4 (MPEG4), Advanced Video Coding (H.264 or AVC), Moving Picture Experts Group Layer-3 (MP3), Advanced Audio Coding (AAC), Adaptive Multi-Rate (AMR) audio codec, Joint Photographic Experts Group (JPEG or JPG), or Portable Network Graphics (PNG)), graphics libraries (e.g., an OpenGL framework used to render in two dimensions

(2D) and three dimensions (3D) in a graphic content on a display), database libraries (e.g., SQLite to provide various relational database functions), web libraries (e.g., WebKit to provide web browsing functionality), and the like. The libraries **1010** can also include a wide variety of other libraries **1028** to provide many other APIs to the applications **1006**.

[0147] The frameworks **1008** provide a common high-level infrastructure that is used by the applications **1006**. For example, the frameworks **1008** provide various graphical user interface (GUI) functions, high-level resource management, and high-level location services. The frameworks **1008** can provide a broad spectrum of other APIs that can be used by the applications **1006**, some of which may be specific to a particular operating system or platform.

[0148] In an example, the applications **1006** may include a home application **1036**, a contacts application **1030**, a browser application **1032**, a book reader application **1034**, a location application **1042**, a media application **1044**, a messaging application **1046**, a game application **1048**, and a broad assortment of other applications such as an external application **1040**. The applications **1006** are programs that execute functions defined in the programs. Various programming languages can be employed to create one or more of the applications **1006**, structured in a variety of manners, such as object-oriented programming languages (e.g., Objective-C, Java, or C++) or procedural programming languages (e.g., C or assembly language). In a specific example, the external application **1040** (e.g., an application developed using the ANDROID™ or IOS™ software development kit (SDK) by an entity other than the vendor of the particular platform) may be mobile software running on a mobile operating system such as IOS™, ANDROID™ WINDOWS® Phone, or another mobile operating system. In this example, the external application **1040** can invoke the API calls **1050** provided by the operating system **1012** to facilitate functionality described herein.

Glossary

[0149] “CARRIER SIGNAL” in this context refers to any intangible medium that is capable of storing, encoding, or carrying transitory or non-transitory instructions for execution by the machine, and includes digital or analog communications signals or other intangible medium to facilitate communication of such instructions. Instructions may be transmitted or received over the network using a transitory or non-transitory transmission medium via a network interface device and using any one of a number of well-known transfer protocols.

[0150] “CLIENT DEVICE” in this context refers to any machine that interfaces to a communications network to obtain resources from one or more server systems or other client devices. A client device may be, but is not limited to, a mobile phone, desktop computer, laptop, PDAs, smart phones, tablets, ultra books, netbooks, laptops, multi-processor systems, microprocessor-based or programmable consumer electronics, game consoles, set-top boxes, or any other communication device that a user may use to access a network.

[0151] “COMMUNICATIONS NETWORK” in this context refers to one or more portions of a network that may be an ad hoc network, an intranet, an extranet, a virtual private network (VPN), a local area network (LAN), a wireless LAN (WLAN), a wide area network (WAN), a wireless WAN (WWAN), a metropolitan area network (MAN), the Internet, a portion of the Internet, a portion of the Public Switched Telephone Network (PSTN), a plain old telephone service (POTS) network, a cellular telephone network, a wireless network, a Wi-Fi® network, another type of network, or a combination of two or more such networks. For example, a network or a portion of a network may include a wireless or cellular network and the coupling may be a Code Division Multiple Access (CDMA) connection, a Global System for Mobile communications (GSM) connection, or other type of cellular or wireless coupling. In this example, the coupling may implement any of a variety of types of data transfer technology, such as Single Carrier Radio Transmission Technology (1×RTT), Evolution-Data Optimized (EVDO) technology, General Packet Radio Service (GPRS) technology, Enhanced Data rates for GSM Evolution (EDGE) technology, third Generation Partnership Project (3GPP) including 3G, fourth generation wireless (4G) networks, Universal

Mobile Telecommunications System (UMTS), High Speed Packet Access (HSPA), Worldwide Interoperability for Microwave Access (WiMAX), Long Term Evolution (LTE) standard, others defined by various standard setting organizations, other long range protocols, or other data transfer technology.

[0152] “EPHEMERAL MESSAGE” in this context refers to a message that is accessible for a time-limited duration. An ephemeral message may be a text, an image, a video, and the like. The access time for the ephemeral message may be set by the message sender. Alternatively, the access time may be a default setting or a setting specified by the recipient. Regardless of the setting technique, the message is transitory.

[0153] “MACHINE-READABLE MEDIUM” in this context refers to a component, device, or other tangible media able to store instructions and data temporarily or permanently and may include, but is not limited to, random-access memory (RAM), read-only memory (ROM), buffer memory, flash memory, optical media, magnetic media, cache memory, other types of storage (e.g., Erasable Programmable Read-Only Memory (EEPROM)) and/or any suitable combination thereof. The term “machine-readable medium” should be taken to include a single medium or multiple media (e.g., a centralized or distributed database, or associated caches and servers) able to store instructions. The term “machine-readable medium” shall also be taken to include any medium, or combination of multiple media, that is capable of storing instructions (e.g., code) for execution by a machine, such that the instructions, when executed by one or more processors of the machine, cause the machine to perform any one or more of the methodologies described herein. Accordingly, a “machine-readable medium” refers to a single storage apparatus or device, as well as “cloud-based” storage systems or storage networks that include multiple storage apparatus or devices. The term “machine-readable medium” excludes signals per se.

[0154] “COMPONENT” in this context refers to a device, physical entity, or logic having boundaries defined by function or subroutine calls, branch points, APIs, or other technologies that provide for the partitioning or modularization of particular processing or control functions. Components may be combined via their interfaces with other components to carry out a machine process. A component may be a packaged functional hardware unit designed for use with other components and a part of a program that usually performs a particular function of related functions. Components may constitute either software components (e.g., code embodied on a machine-readable medium) or hardware components. A “hardware component” is a tangible unit capable of performing certain operations and may be configured or arranged in a certain physical manner. In various example, one or more computer systems (e.g., a standalone computer system, a client computer system, or a server computer system) or one or more hardware components of a computer system (e.g., a processor or a group of processors) may be configured by software (e.g., an application or application portion) as a hardware component that operates to perform certain operations as described herein.

[0155] A hardware component may also be implemented mechanically, electronically, or any suitable combination thereof. For example, a hardware component may include dedicated circuitry or logic that is permanently configured to perform certain operations. A hardware component may be a special-purpose processor, such as a Field-Programmable Gate Array (FPGA) or an ASIC. A hardware component may also include programmable logic or circuitry that is temporarily configured by software to perform certain operations. For example, a hardware component may include software executed by a general-purpose processor or other programmable processor. Once configured by such software, hardware components become specific machines (or specific components of a machine) uniquely tailored to perform the configured functions and are no longer general-purpose processors. It will be appreciated that the decision to implement a hardware component mechanically, in dedicated and permanently configured circuitry, or in temporarily configured circuitry (e.g., configured by software) may be driven by cost and time considerations. Accordingly, the phrase “hardware component” (or “hardware-implemented component”) should

be understood to encompass a tangible entity, be that an entity that is physically constructed, permanently configured (e.g., hardwired), or temporarily configured (e.g., programmed) to operate in a certain manner or to perform certain operations described herein. Considering examples in which hardware components are temporarily configured (e.g., programmed), each of the hardware components need not be configured or instantiated at any one instance in time. For example, where a hardware component comprises a general-purpose processor configured by software to become a special-purpose processor, the general-purpose processor may be configured as respectively different special-purpose processors (e.g., comprising different hardware components) at different times. Software accordingly configures a particular processor or processors, for example, to constitute a particular hardware component at one instance of time and to constitute a different hardware component at a different instance of time.

[0156] Hardware components can provide information to, and receive information from, other hardware components. Accordingly, the described hardware components may be regarded as being communicatively coupled. Where multiple hardware components exist contemporaneously, communications may be achieved through signal transmission (e.g., over appropriate circuits and buses) between or among two or more of the hardware components. In examples in which multiple hardware components are configured or instantiated at different times, communications between such hardware components may be achieved, for example, through the storage and retrieval of information in memory structures to which the multiple hardware components have access. For example, one hardware component may perform an operation and store the output of that operation in a memory device to which it is communicatively coupled. A further hardware component may then, at a later time, access the memory device to retrieve and process the stored output.

[0157] Hardware components may also initiate communications with input or output devices, and can operate on a resource (e.g., a collection of information). The various operations of example methods described herein may be performed, at least partially, by one or more processors that are temporarily configured (e.g., by software) or permanently configured to perform the relevant operations. Whether temporarily or permanently configured, such processors may constitute processor-implemented components that operate to perform one or more operations or functions described herein. As used herein, “processor-implemented component” refers to a hardware component implemented using one or more processors. Similarly, the methods described herein may be at least partially processor-implemented, with a particular processor or processors being an example of hardware. For example, at least some of the operations of a method may be performed by one or more processors or processor-implemented components. Moreover, the one or more processors may also operate to support performance of the relevant operations in a “cloud computing” environment or as a “software as a service” (SaaS). For example, at least some of the operations may be performed by a group of computers (as examples of machines including processors), with these operations being accessible via a network (e.g., the Internet) and via one or more appropriate interfaces (e.g., an API). The performance of certain of the operations may be distributed among the processors, not only residing within a single machine, but deployed across a number of machines. In some example, the processors or processor-implemented components may be located in a single geographic location (e.g., within a home environment, an office environment, or a server farm). In other examples, the processors or processor-implemented components may be distributed across a number of geographic locations.

[0158] “PROCESSOR” in this context refers to any circuit or virtual circuit (a physical circuit emulated by logic executing on an actual processor) that manipulates data values according to control signals (e.g., “commands,” “op codes,” “machine code,” etc.) and which produces corresponding output signals that are applied to operate a machine. A processor may, for example, be a Central Processing Unit (CPU), a Reduced Instruction Set Computing (RISC) processor, a Complex Instruction Set Computing (CISC) processor, a Graphics Processing Unit (GPU), a Digital Signal Processor (DSP), an ASIC, a Radio-Frequency Integrated Circuit (RFIC) or any

combination thereof. A processor may further be a multi-core processor having two or more independent processors (sometimes referred to as “cores”) that may execute instructions contemporaneously.

[0159] “TIMESTAMP” in this context refers to a sequence of characters or encoded information identifying when a certain event occurred, for example giving date and time of day, sometimes accurate to a small fraction of a second.

[0160] Changes and modifications may be made to the disclosed examples without departing from the scope of the present disclosure. These and other changes or modifications are intended to be included within the scope of the present disclosure, as expressed in the following claims.

Modules, Components, and Logic

[0161] Certain examples are described herein as including logic or a number of components, modules, or mechanisms. Modules can constitute either software modules (e.g., code embodied on a machine-readable medium or in a transmission signal) or hardware modules. A “hardware module” is a tangible unit capable of performing certain operations and can be configured or arranged in a certain physical manner. In various examples, one or more computer systems (e.g., a standalone computer system, a client computer system, or a server computer system) or one or more hardware modules of a computer system (e.g., a processor or group of processors) is configured by software (e.g., an application or application portion) as a hardware module that operates to perform certain operations as described herein.

[0162] In some examples, a hardware module is implemented mechanically, electronically, or any suitable combination thereof. For example, a hardware module can include dedicated circuitry or logic that is permanently configured to perform certain operations. For example, a hardware module can be a special-purpose processor, such as a Field-Programmable Gate Array (FPGA) or an Application-Specific Integrated Circuit (ASIC). A hardware module may also include programmable logic or circuitry that is temporarily configured by software to perform certain operations. For example, a hardware module can include software encompassed within a general-purpose processor or other programmable processor. It will be appreciated that the decision to implement a hardware module mechanically, in dedicated and permanently configured circuitry, or in temporarily configured circuitry (e.g., configured by software) can be driven by cost and time considerations.

[0163] Accordingly, the phrase “hardware module” should be understood to encompass a tangible entity, be that an entity that is physically constructed, permanently configured (e.g., hardwired), or temporarily configured (e.g., programmed) to operate in a certain manner or to perform certain operations described herein. As used herein, “hardware-implemented module” refers to a hardware module. Considering examples in which hardware modules are temporarily configured (e.g., programmed), each of the hardware modules need not be configured or instantiated at any one instance in time. For example, where a hardware module comprises a general-purpose processor configured by software to become a special-purpose processor, the general-purpose processor may be configured as respectively different special-purpose processors (e.g., comprising different hardware modules) at different times. Software can accordingly configure a particular processor or processors, for example, to constitute a particular hardware module at one instance of time and to constitute a different hardware module at a different instance of time.

[0164] Hardware modules can provide information to, and receive information from, other hardware modules. Accordingly, the described hardware modules can be regarded as being communicatively coupled. Where multiple hardware modules exist contemporaneously, communications can be achieved through signal transmission (e.g., over appropriate circuits and buses) between or among two or more of the hardware modules. In examples in which multiple hardware modules are configured or instantiated at different times, communications between or among such hardware modules may be achieved, for example, through the storage and retrieval of information in memory structures to which the multiple hardware modules have access. For

example, one hardware module performs an operation and stores the output of that operation in a memory device to which it is communicatively coupled. A further hardware module can then, at a later time, access the memory device to retrieve and process the stored output. Hardware modules can also initiate communications with input or output devices, and can operate on a resource (e.g., a collection of information).

[0165] The various operations of example methods described herein can be performed, at least partially, by one or more processors that are temporarily configured (e.g., by software) or permanently configured to perform the relevant operations. Whether temporarily or permanently configured, such processors constitute processor-implemented modules that operate to perform one or more operations or functions described herein. As used herein, “processor-implemented module” refers to a hardware module implemented using one or more processors.

[0166] Similarly, the methods described herein can be at least partially processor-implemented, with a particular processor or processors being an example of hardware. For example, at least some of the operations of a method can be performed by one or more processors or processor-implemented modules. Moreover, the one or more processors may also operate to support performance of the relevant operations in a “cloud computing” environment or as a “software as a service” (SaaS). For example, at least some of the operations may be performed by a group of computers (as examples of machines including processors), with these operations being accessible via a network (e.g., the Internet) and via one or more appropriate interfaces (e.g., an API).

[0167] The performance of certain of the operations may be distributed among the processors, not only residing within a single machine, but deployed across a number of machines. In some examples, the processors or processor-implemented modules are located in a single geographic location (e.g., within a home environment, an office environment, or a server farm). In other examples, the processors or processor-implemented modules are distributed across a number of geographic locations.

Claims

1. A method comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.

2. The method of claim 1, wherein the first virtual coordinate comprises a default position for the one or more virtual objects.

3. The method of claim 2, further comprising: determining a type associated with the one or more virtual objects; and obtaining the default position based on the type associated with the one or more virtual objects.

4. The method of claim 3, further comprising: associating the second virtual coordinate with the type associated with the one or more virtual objects to replace the default position with the second virtual coordinate.

5. The method of claim 1, further comprising: computing a virtual offset between the second virtual coordinate of the one or more virtual objects and a point on the user device, the virtual offset being defined by a vertical height relative to a surface of the real-world environment and a virtual distance between the point and the second virtual coordinate; and storing the virtual offset in a profile associated with a user.

6. The method of claim 5, further comprising: receiving an additional request to access an additional virtual object; retrieving a default virtual position of the additional virtual object; computing a target virtual position by adjusting the default virtual position based on the virtual

offset; and displaying the additional virtual object at the target virtual position.
7. The method of claim 6, further comprising: displaying the additional virtual object at a same vertical height relative to the surface of the real-world environment and a same virtual distance between the point of the user device and the second virtual coordinate as the vertical height and the virtual distance of the one or more virtual objects.
8. The method of claim 1, further comprising: receiving input that selects and moves the one or more virtual objects to a third virtual coordinate in 3D space that is outside of a field of view (FOV) of the user device.
9. The method of claim 8, further comprising: in response to receiving the input that selects and moves the one or more virtual objects to the third virtual coordinate in 3D space that is outside of the FOV of the user device, automatically positioning the one or more virtual objects at a predetermined virtual coordinate in 3D space, the predetermined virtual coordinate corresponding to a position that is inside of the FOV of the user device.
10. The method of claim 9, further comprising: receiving input that defines the predetermined virtual coordinate prior to receiving the input that moves the one or more virtual objects to the third virtual coordinate.
11. The method of claim 9, wherein the one or more virtual objects are automatically positioned at the predetermined virtual coordinate in 3D space in response to receiving a command to terminate moving of the one or more virtual objects to the third virtual coordinate in 3D space that is outside of the field of view (FOV) of the user device.
12. The method of claim 1, wherein the user device comprises an augmented reality (AR) eyewear device.
13. The method of claim 1, wherein the one or more objects comprise one or more thumbnails representing respective media assets.
14. The method of claim 1, wherein the one or more objects comprise conversation elements that represent a conversation in which a user of the user device is involved.
15. The method of claim 1, wherein the one or more objects comprise user interface elements for controlling and playing back one or more content items.
16. The method of claim 1, wherein the one or more objects comprise user interface elements for generating augmented reality content.
17. The method of claim 1, wherein receiving the request comprises: detecting by one or more cameras of the user device a position of hands of the user at the first virtual coordinate; determining that the hands of the user perform a gesture associating with moving the one or more virtual objects; and determining that the hands have been moved to the second virtual coordinate.
18. A system comprising: a storage device; and at least one processor configured to perform operations comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.
19. The system of claim 18, wherein the first virtual coordinate comprises a default position for the one or more virtual objects.
20. A non-transitory machine-readable storage medium comprising instructions that, when executed by one or more processors of a machine, cause the machine to perform operations comprising: displaying, by at least one processor on a first portion of a real-world environment visible on a display of a user device, one or more virtual objects at a first virtual coordinate in three-dimensional (3D) space; receiving a request to move the one or more virtual objects to a second virtual coordinate in 3D space; and in response to receiving the request, maintaining display of the

one or more virtual objects at the second virtual coordinate as a second portion of the real-world environment is visible on the display of the user device.