(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
30 December 2020 (30.12.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/264551 A2

(51) International Patent Classification:
G06T 19/00 (2011.01) *G06T 15/04* (2011.01)

(21) International Application Number:
PCT/US2020/070158

(22) International Filing Date:
19 June 2020 (19.06.2020)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
16/457,461 28 June 2019 (28.06.2019) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 16/457,461 (CON)
Filed on 28 June 2019 (28.06.2019)

(71) Applicant: **SNAP INC.** [US/US]; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US).

(72) Inventors: **HARE, Samuel Edward**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US). **MCPHEE, Andrew James**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US). **LAZAROV, Maxim Maximov**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US). **SHANG, Wentao**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US). **GOODRICH, Kyle**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US). **MATHEW, Tony**; 2772 Donald Douglas Loop North, Santa Monica, California 90405 (US).

(74) Agent: **PERDOK, Monique M.** et al.; Schwegman Lundberg & Woessner, P.A., P.O. Box 2938, Minneapolis, Minnesota 55402 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) Title: 3D OBJECT CAMERA CUSTOMIZATION SYSTEM

(57) Abstract: Systems and methods are provided for capturing by a camera of a user device, a first image depicting a first environment of the user device; overlaying a first virtual object on a portion of the first image depicting the first environment; modifying a surface of the first virtual object using content captured by the user device; storing a second virtual object comprising the first virtual object with the modified surface; and generating for display the second virtual object on a portion of a second image depicting a second environment.

WO 2020/264551 A2

3D OBJECT CAMERA CUSTOMIZATION SYSTEM

CLAIM OF PRIORITY

[0001] This application claims priority to U.S. Patent Application Serial No. 16/457,461, filed on June 28, 2019, which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] The present disclosure relates generally to visual presentations of virtual content and, more particularly, to rendering a three-dimensional object within a real-world environment captured in a camera feed of a computing device.

BACKGROUND

[0003] Virtual rendering systems can be used to create engaging and entertaining augmented reality experiences, in which three-dimensional (3D) virtual object graphics content appear to be present in the real-world. Such systems allow a user to select from a predefined list of 3D objects and display the selected 3D objects in the view of a camera feed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] In the drawings, which are not necessarily drawn to scale, like numerals may describe similar components in different views. To easily identify the discussion of any particular element or act, the most significant digit or digits in a reference number refer to the figure number in which that element is first introduced. Some embodiments are illustrated by way of example, and not limitation, in the figures of the accompanying drawings in which:

[0005] FIG. 1 is a block diagram showing a messaging system for exchanging data (e.g., messages and associated content) over a network, according to example embodiments.

[0006] FIG. 2 is block diagram illustrating further details regarding a messaging system, according to example embodiments.

[0007] FIG. 3 is a schematic diagram illustrating data which may be stored in the database of the messaging server system, according to example embodiments.

[0008] FIG. 4 is a schematic diagram illustrating a structure of a message generated by a messaging client application for communication, according to example embodiments.

[0009] FIG. 5 is a block diagram illustrating various components of a surface environment representation object generation system, which may be provided as part of the messaging system, according to example embodiments.

[0010] FIGS. 6 and 7 are flowcharts illustrating example operations of the surface environment representation object generation system in performing a method for generating a message that includes a surface environment representation object, according to example embodiments.

[0011] FIGS. 8-12A and 12B are interface diagrams that illustrate various interfaces provided by the messaging system, according to some example embodiments.

[0012] FIG. 13 is a block diagram illustrating a representative software architecture, which may be used in conjunction with various hardware architectures herein described, according to example embodiments.

[0013] FIG. 14 is a block diagram illustrating components of a machine able to read instructions from a machine-readable medium (e.g., a machine-readable storage medium) and perform any one or more of the methodologies discussed herein, according to example embodiments.

DETAILED DESCRIPTION

[0014] The description that follows includes systems, methods, techniques, instruction sequences, and computing machine program products that embody illustrative embodiments 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 embodiments of the inventive subject matter. It will be evident, however, to those skilled in the art, that embodiments of the inventive 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.

[0015] Users are always seeking new ways to interact with content as if it exists in the real world. Typical augmented reality systems enable a user to insert a virtual object into a video frame and control its position in the real-world environment depicted in the video frame. However, simply inserting a virtual object into a video frame without considering other objects (e.g., surfaces) in the video frame makes the resulting video frame with the virtual object look un-realistic, particularly because depth of the object cannot be utilized. Some systems allow a user to choose from a predefined list of virtual objects to insert into a real-word environment that is depicted in a camera feed (e.g., a live video received from a camera). While such systems generally work well in presenting such objects in the real-world environment, the lack of ability to customize and manipulate the objects makes the systems less appealing and less interesting to users.

[0016] Among other things, embodiments of the present disclosure improve the functionality of electronic messaging and imaging software and systems by providing functionality that allows users to modify surface textures of a virtual object, such as a three-dimensional (3D) caption or two-dimensional (2D) caption, user generated content, pre-generated content, curated content, geofilter, picture, lens, graphical objects, stickers, emojis, and a GIFs to represent portions of an image depicting a first real-world environment. Specifically, portions of the virtual object a user inserts can be transparent or partially transparent to reveal and represent the underlying images of the

real-world environment and/or modified to include other images of the real-world environment, such as using additional camera feeds. The virtual objects having the modified surface textures representing the portions of the image depicting the first real-world environment are then rendered, as if they exist in other real-world environments and/or in other viewpoints of the real-world environment. In this way, the disclosed embodiments create a new and more engaging ways for users to communicate with each other. Namely, rather than exchanging messages using predetermined objects that the user cannot manipulate or personalize, the disclosed embodiments allow users to create virtual objects having surface environment representations with personalized content (e.g., messages).

[0017] The disclosed embodiments allow the users to modify the virtual objects such that the graphical properties (e.g., the visual attributes, such as colors and textures) of the created virtual objects represent the real-world environment of the user who is creating the message. This enables the user to convey information to other users using messaging that have enriched content. The graphical user interface functions, according to the disclosed embodiments, enable easy generation of an environment-based virtual object from within a messaging application, and without requiring sophisticated photo-editing skills from the user.

[0018] In some embodiments, a first real-world environment content item, such as images and/or video of a sandy beach, the ocean and the sky, generated by an image-capturing device (e.g., a digital camera) is displayed and 2D/3D text can be displayed in on top of a portion of the first real-world environment content, such as on top of the beach. The surface of the text (e.g., the colors of the characters of the text) is modified using the colors and patterns of the portion of the first real-world environment over which the text is displayed to appear as if the text is at least partially transparent, revealing the underlying real-world environment. Specifically, a surface texture of one character of the text that is displayed over the sandy beach is modified to be at least partially transparent in a way that represents the color and pattern of the sand (e.g., sandy white) and a surface texture of another

character of the text that is displayed over the ocean is similarly modified to represent the color of the ocean (e.g., a wavy blue). The text with the modified surface texture is stored and sent to another user for presentation on top of a second real-world environment depicted in an image, such as an environment depicting an office scene. This way, the text with the mixture of a sandy white beach and ocean colored characters of the first real-world environment is presented and can be moved around in an image depicting the office scene in the second real-world environment. According to the disclosed embodiments, a first user who is located at the first real-world environment (e.g., the beach) can convey a textual message to a second user, who may be in the office (another real-world environment) that is enriched with visual attributes of the first user's surroundings (e.g., the beach). This significantly enhances the meaning of the message the first user is trying to convey to the second user.

[0019] FIG. 1 is a block diagram showing an example messaging system 100 for exchanging data (e.g., messages and associated content) over a network. The messaging system 100 includes multiple client devices 102, each of which hosts a number of applications including a messaging client application 104. Each messaging client application 104 is communicatively coupled to other instances of the messaging client application 104 and a messaging server system 108 via a network 106 (e.g., the Internet).

[0020] Accordingly, each messaging client application 104 is able to communicate and exchange data with another messaging client application 104 and with the messaging server system 108 via the network 106. The data exchanged between messaging client applications 104, and between a messaging client application 104 and the messaging 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).

[0021] In some embodiments, the messaging client application 104 presents a graphical user interface to a user for selecting or creating a virtual surface environment representation object, such as a 3D surface environment representation object and/or a 2D surface environment representation object.

A user can activate a camera of the messaging client application 104 to view in real-time images of the user's real-world surroundings/environment (e.g., the camera feed). The user can instruct the messaging client application 104 to add a virtual object to real-world images being captured by the camera at a user selected position. The position can be selected in 3D and/or 2D space. Specifically, the user can manipulate the virtual object to reposition the virtual object relative to the real-world objects.

[0022] In some embodiments, the messaging client application 104 receives user input to modify the added virtual object to be a virtual surface environment representation object. Namely, after or before the user positions the virtual object within the real-world environment depicted in the real-time images, the user can tap on the screen or make another suitable gesture to instruct the messaging client application 104 to modify the surface texture of the virtual object using the underlying visual attributes of the real-world environment over which the virtual object is positioned. In an embodiment, the messaging client application 104 changes a transparency of the surface pixels of the virtual object such that pixels within a border of the virtual object represent the underlying portions of the image of the real-world environment over which the virtual object is positioned. The user can then instruct the messaging client application 104 to change the position of the virtual object in 2D/3D space. As the virtual object with the modified surface texture is repositioned, the surface texture can continuously be updated to continue representing the underlying portions of the image of the real-world over which the object is positioned in real-time. In this way, as the user moves the object around, all pixels within the border of the object continue to show in a partially transparent manner the pixels of the real-world environment over which the pixels within the border of the object are positioned.

[0023] After the object is placed in 2D/3D space over the image portions depicting the real-world environment, the user can instruct the messaging client application 104 to capture and store the surface textures of the virtual object in their present state. Namely, the pixels of the surface textures of the

virtual object are maintained at their present value depicting in a partially transparent manner portions of the real-world environment over which they are positioned. After the messaging client application 104 captures and stores the current state of the surface textures, the virtual object becomes locked in a way that allows the user to continue manipulating the position in 2D/3D space within the real-world environment depicted in the camera feed. But now, as the locked virtual object is moved around, the surface textures no longer are updated and modified to represent the underlying images of the real-world environment.

[0024] As an example, the real-world environment depicted in the camera feed includes the ocean, a sandy beach, trees and the sky. A user can position a 2D/3D virtual object, such as a caption with text, within the real-world environment. Specifically, a first set of letters of the virtual object can be positioned over the ocean and a second set of letters of the virtual object can be positioned over the sandy beach. The messaging client application 104 can be instructed to modify the surface textures of the virtual object to generate a surface environment representation object. In response, the first set of letters can become partially transparent to reveal the ocean over which they are positioned (e.g., the pixel values over which the first set of letters are positioned are copied into the pixel vales of the pixels within a border of the first set of letters). Similarly, the second set of letters can become partially transparent to reveal the sandy beach over which they are positioned (e.g., the pixel values over which the second set of letters are positioned are copied into the pixel vales of the pixels within a border of the second set of letters). The user can manipulate the position of the caption and the pixel values within the borders of the letters continue to be updated to represent (e.g., show, display, have an appearance based on) the underlying image portions over which they are positioned in a partially transparent manner.

[0025] In some embodiments, a surface texture/appearance of the virtual object can be generated to simulate a reflective (e.g., mirror-like) surface located in the first environment (e.g., instead of being based on partial

transparency. For example, the surface texture/appearance of the virtual object can represent content items within the environment that are in front of the virtual object (rather than behind the virtual object) relative to the virtual object's position in the 3D space.

[0026] The user can instruct the messaging client application 104 to lock the current state of the virtual object. In response, as the user moves the virtual object around the real-world environment, the surface texture and pixel values are no longer updated. In this way, the first set of letters, with the surface texture of the ocean, can be moved to be positioned over the beach within the real-world environment but continue to show the surface texture of the ocean. Also, the second set of letters, with the surface texture of the beach, can be moved to be positioned over the sky within the real-world environment but continue to show the surface texture of the beach. Specifically, the first and second set of letters are no longer in a partially transparent state and can be moved around to other portions of the real-world environment in a way that obscures the portions over which they are positioned using the pixel values of the letters when the letters were positioned last over the portion of the environment when the letters were in a partially transparent state.

[0027] In some embodiments, the user can capture, store and share the virtual surface environment representation object with another user of another messaging client application 104. For example, a second user who is within a real-world office environment can activate a camera feed. The second user can add the received virtual surface environment representation object into a camera feed depicting real-time images of the second user's surroundings (e.g., the office environment). In some cases, the received virtual surface environment representation object is automatically placed into the camera feed when the object is received by the second user. Namely, upon the second user receiving the virtual surface environment representation object from another user, the camera feed of the second user is automatically activated and displays the received virtual surface environment representation object. In some cases, the second user receives

the virtual surface environment representation object by way of a message (which may or may not include an indication that the message includes a virtual surface environment representation object). In response to the second user selecting the message (e.g., the user opens the message to view its contents), the virtual surface environment representation object is retrieved and displayed automatically in the camera feed of the second user. In this way, the second user can view the text, generated by the first user while the first user was at the beach environment and with the surface textures of the beach environment, in the office environment. The second user can move around the surface environment representation object within the office environment depicted in the real-time images. This results in the second user viewing characters that show partially images of the ocean and the sand from the first user's real-world environment within the images of the second user's environment.

[0028] In some embodiments, the user can instruct the messaging client application 104 to add a virtual object, such as a hat, to the user's real-world environment depicted in real-time images captured by an initial camera feed of the user's device. For example, the real-world environment depicted in the images includes a head of a person. The user can instruct the messaging client application 104 to add a virtual object, such as a hat, to the real-world environment and the user can position the virtual object on top of the head. Specifically, the messaging client application 104 can place the virtual hat on top of the person's head in the real-world environment. The user can activate the surface environment representation lens feature of the messaging client application 104 and can select a portion of the virtual object to be modified. For example, the user can draw a shape to be modified to represent the user's real-world environment and/or the messaging client application 104 can modify predetermined portions of the virtual object, such as the front of the hat, to represent the real-world environment. In an embodiment, the messaging client application 104 can activate another camera, such as another front facing camera or rear facing camera. The messaging client application 104 can receive real-time images of the

activated camera and insert the received images into the portion of the virtual object selected to be modified to represent the real-world environment.

[0029] After the virtual object is placed in 2D/3D space over the image portions depicting the real-world environment and after the user is satisfied with the real-time images received from the additional camera that are presented on the virtual object, the user can instruct the messaging client application 104 to capture and store the surface textures of the virtual object in their present state. Namely, the currently received image from the additional camera is stored and maintained in the position on the virtual object. The user to continue manipulating the position in 2D/3D space within the real-world environment depicted in the initial camera feed. But now, as the locked virtual object is moved around, the surface textures no longer are updated and modified to represent the images of the additional camera. For example, the front of the virtual hat depicting the last image received from the additional camera feed can be moved around to other portions of the real-world environment depicted in the initial camera feed. The user can also send this virtual object to another user for presentation and manipulation in the second user's environment or a different camera feed.

[0030] The messaging server system 108 provides server-side functionality via the network 106 to a particular messaging client application 104. While certain functions of the messaging system 100 are described herein as being performed by either a messaging client application 104 or by the messaging server system 108, it will be appreciated that the location of certain functionality either within the messaging client application 104 or the messaging server system 108 is a design choice. For example, it may be technically preferable to initially deploy certain technology and functionality within the messaging server system 108, but to later migrate this technology and functionality to the messaging client application 104 where a client device 102 has a sufficient processing capacity.

[0031] The messaging server system 108 supports various services and operations that are provided to the messaging client application 104. Such

operations include transmitting data to, receiving data from, and processing data generated by the messaging client application 104. This data may include message content, client device information, geolocation information, media annotation and overlays, message content persistence conditions, social network information, and live event information, as examples. Data exchanges within the messaging system 100 are invoked and controlled through functions available via user interfaces (UIs) of the messaging client application 104.

[0032] Turning now specifically to the messaging server system 108, an Application Program Interface (API) server 110 is coupled to, and provides a programmatic interface to, an application server 112. The application server 112 is communicatively coupled to a database server 118, which facilitates access to a database 120 in which is stored data associated with messages processed by the application server 112.

[0033] Dealing specifically with the API server 110, this server receives and transmits message data (e.g., commands and message payloads) between the client device 102 and the application server 112. Specifically, the API server 110 provides a set of interfaces (e.g., routines and protocols) that can be called or queried by the messaging client application 104 in order to invoke functionality of the application server 112. The API server 110 exposes various functions supported by the application server 112, including account registration, login functionality, the sending of messages, via the application server 112, from a particular messaging client application 104 to another messaging client application 104, the sending of media files (e.g., images or video) from a messaging client application 104 to the messaging server application 114, and for possible access by another messaging client application 104, the setting of a collection of media data (e.g., story), the retrieval of such collections, the retrieval of a list of friends of a user of a client device 102, the retrieval of messages and content, the adding and deleting of friends to a social graph, the location of friends within a social graph, opening an application event (e.g., relating to the messaging client application 104).

[0034] The application server 112 hosts a number of applications and subsystems, including a messaging server application 114, an image processing system 116, and a social network system 122. The messaging server application 114 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 messaging client 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, by the messaging server application 114, to the messaging client application 104. Other processor and memory intensive processing of data may also be performed server-side by the messaging server application 114, in view of the hardware requirements for such processing.

[0035] The application server 112 also includes an image processing system 116 that is dedicated to performing various image processing operations, typically with respect to images or video received within the payload of a message at the messaging server application 114.

[0036] The social network system 122 supports various social networking functions and services, and makes these functions and services available to the messaging server application 114. To this end, the social network system 122 maintains and accesses an entity graph within the database 120. Examples of functions and services supported by the social network system 122 include the identification of other users of the messaging 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.

[0037] The application server 112 is communicatively coupled to a database server 118, which facilitates access to a database 120 in which is stored data associated with messages processed by the messaging server application 114.

[0038] FIG. 2 is block diagram illustrating further details regarding the messaging system 100, according to example embodiments. Specifically, the messaging system 100 is shown to comprise the messaging client application 104 and the application server 112, which in turn embody a number of some subsystems, namely an ephemeral timer system 202, a collection management system 204, and an annotation system 206.

[0039] The ephemeral timer system 202 is responsible for enforcing the temporary access to content permitted by the messaging client application 104 and the messaging server application 114. To this end, 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 display and enable access to messages and associated content via the messaging client application 104.

[0040] The collection management system 204 is responsible for managing collections of media (e.g., collections of text, image, video, and audio data). In some examples, 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 messaging client application 104.

[0041] The collection management system 204 furthermore includes a curation interface 208 that allows a collection manager to manage and curate a particular collection of content. For example, the curation interface 208 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 embodiments, compensation may be

paid to a user for inclusion of user generated content into a collection. In such cases, the curation interface 208 operates to automatically make payments to such users for the use of their content.

[0042] The annotation system 206 provides various functions that enable a user to annotate or otherwise modify or edit media content associated with a message. For example, the annotation system 206 provides functions related to the generation and publishing of media overlays for messages processed by the messaging system 100. The annotation system 206 operatively supplies a media overlay (e.g., a filter or lens) to the messaging client application 104. In another example, the annotation system 206 operatively supplies a media overlay to the messaging client application 104 based on other information, such as social network information of the user of the client 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.

[0043] The audio and visual content or the visual effects can be applied to a media content item (e.g., a photo) at the client device 102. For example, the media overlay including text that can be overlaid on top of an image or video generated by the client 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).

[0044] The annotation system 206 includes a surface environment representation object generation system 210 that provides functionality to receive a virtual object (e.g., a virtual object with transparency type of surface environment representation or a virtual object with a region in which additional camera feeds are presented) and generate a virtual surface environment representation object, from the received virtual object, and display and track the virtual surface environment representation object, at positions relative to the client device 102, within a 2D/3D space captured within a camera feed of the client device 102 (also referred to by those of ordinary skill in the art as a "camera stream," "a video stream," or a "video

feed"). The virtual surface environment representation objects generated, displayed, and tracked by the surface environment representation object generation system 210 include virtual surface environment representation objects generated and modified to represent portions of a real-world environment over which the objects are positioned in an image depicting the real-world environment. A virtual surface environment representation object represents any user generated content, curated content, images, text, video, captions, animation or other visual item selected by a user or automatically identified that has surface textures that depict image portions captured by one or more cameras of a user device.

[0045] The surface environment representation object generation system 210 provides functionality to enable users to author, edit, and preview virtual surface environment representation objects by supplying a virtual object and positioning the object over a desired portion of an image that depicts a real-world environment. The real-world environment can be a pre-recorded video, a pre-captured image, a real-time video, or a real-time image. To this end, the surface environment representation object generation system 210 includes an editing interface 212 and a preview interface 214. The editing interface 212 allows a user to author and select a virtual object (e.g., the user can select a 2D/3D image, 2D/3D graphical element or avatar, or video from a list or can manually type in a set of characters corresponding to text or a caption).

[0046] The editing interface 212 enables users to edit the virtual object using keyboard input and other types of input including touch screen based gestures. For example, the user can change the scale, color scheme, size or any other visual attribute of the virtual object. After the user is satisfied with the edited virtual object, the user can instruct the surface environment representation object generation system 210 to create a virtual surface environment representation object from the virtual object input by the user. The preview interface 214 allows a user to preview and review the generated virtual surface environment representation object before generating a message that includes the virtual surface environment representation object.

The preview interface 214 may also enable the user to edit the presentation of the virtual surface environment representation object (e.g., by changing a scale, orientation, or placement in 2D/3D space of the virtual surface environment representation object on the display screen). The surface environment representation object generation system 210 creates the virtual surface environment representation object from the virtual object selected by the user in accordance with the process described in connection with FIGS. 6 and 7.

[0047] The surface environment representation object generation system 210 may cause a virtual surface environment representation object to be displayed (e.g., on a display of the client device 102) at positions in a 3D space captured within the camera feed based on a reference surface (e.g., the ground) detected in the 3D space. As will be discussed in further detail below, the surface environment representation object generation system 210 comprises a redundant tracking system comprising a set of tracking subsystems configured to track a virtual surface environment representation object at positions in 2D/3D space based on a set of tracking indicia, and transition between tracking subsystems. The surface environment representation object generation system 210 may further transition between tracking with six degrees of freedom (6DoF) and tracking with three degrees of freedom (3DoF) based on an availability of the tracking indicia.

[0048] FIG. 3 is a schematic diagram 300 illustrating data, which may be stored in the database 120 of the messaging server system 108, according to certain example embodiments. While the content of the database 120 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).

[0049] The database 120 includes message data stored within a message table 314. An entity table 302 stores entity data, including an entity graph 304. Entities for which records are maintained within the entity table 302 may include individuals, corporate entities, organizations, 2D and/or 3D objects, 3D object templates, 3D object textures, places, events, virtual

objects configured to have at least a predetermined portion be in a transparent, partially transparent, or locked state, virtual objects configured with selectable or predetermined portions for representing additional camera feeds, and so forth. Regardless of type, any entity regarding which the messaging 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).

[0050] The entity graph 304 furthermore stores information regarding relationships and associations between entities. Such relationships may be social, professional (e.g., work at a common corporation or organization), interested-based, or activity-based, merely for example.

[0051] The database 120 also stores annotation data, in the example form of filters and lenses, in an annotation table 312. Filters and lens for which data is stored within the annotation table 312 are associated with and applied to videos (for which data is stored in a video table 310) and/or images (for which data is stored in an image table 308). Filters are overlays that are displayed as overlaid on an image or video during presentation to a recipient user. Lens, on the other hand, include real-time special effect and/or sounds that may be added to images of a camera feed while a user is composing a message. In comparison, filters are applied to an image or video after the image or video is captured at the client device 102 while a lens is applied to the camera feed of the client device 102 such that when an image or videos is captured at the client device 102 with a lens applied, the applied lens is incorporated as part of the image or video that is generated. Filters and lenses may be of various types, including user-selected filters and lens from a gallery of filters or a gallery of lenses presented to a sending user by the messaging client application 104 when the sending user is composing a message. Any functionality described with respect to filters is equally applicable to lens and vice versa.

[0052] As mentioned above, the video table 310 stores video data which, in one embodiment, is associated with messages for which records are maintained within the message table 314. Similarly, the image table 308

stores image data associated with messages for which message data is stored in the entity table 302. The entity table 302 may associate various annotations from the annotation table 312 with various images and videos stored in the image table 308 and the video table 310.

[0053] A story table 306 stores data regarding collections of messages and associated image, 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 302). 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 UI of the messaging client 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.

[0054] 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 messaging client application 104, to contribute content to a particular live story. The live story may be identified to the user by the messaging client application 104, based on his or her location. The end result is a "live story" told from a community perspective.

[0055] A further type of content collection is known as a "location story," which enables a user whose client 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 embodiments, 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).

[0056] FIG. 4 is a schematic diagram illustrating a structure of a message 400, according to some in some embodiments, generated by a messaging client application 104 for communication to a further messaging client application 104 or the messaging server application 114. The content of a particular message 400 is used to populate the message table 314 stored within the database 120, accessible by the messaging server application 114. Similarly, the content of a message 400 is stored in memory as "in-transit" or "in-flight" data of the client device 102 or the application server 112. The message 400 is shown to include the following components:

- A message identifier 402: a unique identifier that identifies the message 400.
- A message text payload 404: text, to be generated by a user via a user interface of the client device 102 and that is included in the message 400.
- A message image payload 406: image data, captured by a camera component of a client device 102 or retrieved from memory of a client device 102, and that is included in the message 400.
- A message video payload 408: video data, captured by a camera component or retrieved from a memory component of the client device 102 and that is included in the message 400.
- A message audio payload 410: audio data, captured by a microphone or retrieved from the memory component of the client device 102, and that is included in the message 400.
- A message annotations 412: annotation data (e.g., filters, stickers or other enhancements) that represents annotations to be applied to message image payload 406, message video payload 408, or message audio payload 410 of the message 400.
- A message duration parameter 414: parameter value indicating, in seconds, the amount of time for which content of the message (e.g., the message image payload 406, message video payload 408, message audio payload 410) is to be presented or made accessible to a user via the messaging client application 104.

- A message geolocation parameter 416: geolocation data (e.g., latitudinal and longitudinal coordinates) associated with the content payload of the message. Multiple message geolocation parameter 416 values may be included in the payload, with each of these parameter values being associated with respect to content items included in the content (e.g., a specific image into within the message image payload 406, or a specific video in the message video payload 408).
- A message story identifier 418: identifier value identifying one or more content collections (e.g., "stories") with which a particular content item in the message image payload 406 of the message 400 is associated. For example, multiple images within the message image payload 406 may each be associated with multiple content collections using identifier values.
- A message tag 420: each message 400 may be tagged with multiple tags, each of which is indicative of the subject matter of content included in the message payload. For example, where a particular image included in the message image payload 406 depicts an animal (e.g., a lion), a tag value may be included within the message tag 420 that is indicative of the relevant animal. Tag values may be generated manually, based on user input, or may be automatically generated using, for example, image recognition.
- A message sender identifier 422: an identifier (e.g., a messaging system identifier, email address, or device identifier) indicative of a user of the client device 102 on which the message 400 was generated and from which the message 400 was sent.
- A message receiver identifier 424: an identifier (e.g., a messaging system identifier, email address or device identifier) indicative of a user of the client device 102 to which the message 400 is addressed.

[0057] The contents (e.g. values) of the various components of message 400 may be pointers to locations in tables within which content data values are stored. For example, an image value in the message image payload 406 may be a pointer to (or address of) a location within an image table 308.

Similarly, values within the message video payload 408 may point to data stored within a video table 310, values stored within the message annotations 412 may point to data stored in an annotation table 312, values stored within the message story identifier 418 may point to data stored in a story table 306, and values stored within the message sender identifier 422 and the message receiver identifier 424 may point to user records stored within an entity table 302.

[0058] FIG. 5 is a block diagram illustrating functional components of the surface environment representation object generation system 210 that configures the surface environment representation object generation system 210 to render surface environment representation objects from virtual objects in a 2D/3D space depicted in a live camera feed. The surface environment representation object generation system 210 is shown as including a rendering module 502, a tracking module 504, and a disruption detection module 506. The various components and modules of the surface environment representation object generation system 210 may be configured to communicate with each other (e.g., via a bus, shared memory, or a switch). Although not illustrated in FIG. 5, in some embodiments, the surface environment representation object generation system 210 may include or may be in communication with one or more cameras (e.g., one or more front-facing cameras and/or one or more rear-facing cameras) configured to produce a camera feed comprising image data that includes a sequence of images (e.g., a video). In an embodiment, depending on the type of virtual object selected by the user, the surface environment representation object generation system 210 may activate multiple of the cameras on the user device. A first camera may be activated to display real-time images of the user's real-world environment in full screen and a second camera may be activated to display real-time images of the user's real-world environment on a dedicated or user selected portion of a virtual object positioned within the feed from the first camera.

[0059] Any one or more of the components and modules described may be implemented using hardware alone (e.g., one or more of the processors 1404

of a machine) or a combination of hardware and software. For example, any component and modules described of the surface environment representation object generation system 210 may physically include an arrangement of one or more of the processors 1404 (e.g., a subset of or among the one or more processors of the machine) configured to perform the operations described herein for that component. As another example, any component and module of the surface environment representation object generation system 210 may include software, hardware, or both, that configure an arrangement of one or more processors 1404 (e.g., among the one or more processors of the machine) to perform the operations described herein for that component. Accordingly, different components and modules of the surface environment representation object generation system 210 may include and configure different arrangements of such processors 1404 or a single arrangement of such processors 1404 at different points in time.

[0060] Moreover, any two or more components and modules of the surface environment representation object generation system 210 may be combined into a single component or module, and the functions described herein for a single component or module may be subdivided among multiple components and modules. Furthermore, according to various example embodiments, components and modules described herein as being implemented within a single machine, database, or device may be distributed across multiple machines, databases, or devices.

[0061] The tracking system 504 may comprise a first tracking sub-system 504A, a second tracking sub-system 504B, and a third tracking sub-system 504C. Each tracking sub-system tracks the position of a surface environment representation object within the 2D/3D space based on a set of tracking indicia.

[0062] Tracking systems are subject to frequent tracking failure due to environmental conditions, user actions, unanticipated visual interruption between camera and object/scene being tracked, and so forth. Traditionally, such tracking failures would cause a disruption in the presentation of virtual objects in a 3D space. For example, a virtual object may disappear or

otherwise behave erratically, thereby interrupting the illusion of the virtual object being presented within the 3D space. This undermines the perceived quality of the 3D experience as a whole.

[0063] Traditional tracking systems rely on a single approach (Natural Feature Tracking (NFT), Simultaneous Localization and Mapping (SLAM), Gyroscopic, etc.) that each have breaking points in real-world usage due to inaccurate sensor data, movement, loss or occlusion of visual marker, or dynamic interruptions to a scene. Further, each approach may have individual limitations in capability. For example, a gyroscopic tracking system can only track items with 3DoF. Further, utilization of a single tracking system provides inaccurate or unstable position estimation, due to inherent limitations of each individual system. For example, an NFT system may not provide sufficient pitch, yaw, or roll estimation due to the inaccuracies of visual tracking alone, while gyroscopic tracking systems provide inaccurate translation (up, down, left, right).

[0064] To address the foregoing issues with traditional tracking systems, the surface environment representation object generation system 210 comprises multiple redundant tracking sub-systems 504A-C that enable seamless transitions between tracking sub-systems. The multiple redundant tracking sub-systems 504A-C address the issues with traditional tracking systems by merging multiple tracking approaches into a single tracking system 504. The tracking system 504 is able to combine 6DoF and 3DoF tracking techniques through combining and transitioning between multiple tracking systems based on the availability of tracking indicia (e.g., roll, pitch, yaw, natural features, etc.) tracked by the tracking systems. Thus, as the indicia tracked by any one tracking system becomes unavailable, the surface environment representation object generation system 210 seamlessly switches between tracking in 6DoF and 3DoF, thereby providing the user with an uninterrupted experience. For example, in the case of visual tracking systems (e.g., NFT, SLAM), tracking indicia typically analyzed to determine orientation may be replaced with gyroscopic tracking indicia from a gyroscopic tracking system. This would thereby enable transitioning

between tracking in 6Dof and 3DoF based on the availability of tracking indicia.

[0065] In some example embodiments, to transition between tracking in 6DoF and 3DoF, the surface environment representation object generation system 210 gathers and stores tracking indicia within a tracking matrix that includes translation indicia (e.g., up, down, left, right) and rotation indicia (e.g., pitch, yaw, roll). The translation indicia gathered by an NFT system may thereby be extracted from the tracking matrix and utilized when future translation indicia gathered by the NFT system become inaccurate or unavailable. In the meantime, the rotation indicia continue to be provided by the gyroscope. In this way, when the mobile device loses tracking indicia, the tracked objects that are presented in the 3D space will not be changed abruptly at the frame when the tracking indicia are lost. Subsequently, when the target tracking object reappears in the screen, and a new translation T_1 is obtained, the translation part of the view matrix will then be taking advantage of the new translation T_1, and use T_1-T_0 as the translation of the view matrix.

[0066] The rendering module 502 of the surface environment representation object generation system 210 is configured to obtain a virtual object and generate and render surface environment representation objects from the obtained virtual object in a 3D space captured within a live camera feed produced by a camera. In rendering the surface environment representation object, the surface environment representation object generation system 210 assigns the surface environment representation object to a position in the 2D/3D space based on a real-world reference surface detected in the 2D/3D space.

[0067] The surface environment representation object generation system 210 may thereafter track the position of the surface environment representation object relative to a user device in the 3D space by one or more tracking systems in 6DoF. For example, the one or more tracking systems of the surface environment representation object generation system 210 may collect and analyze a set of tracking indicia (e.g., roll, pitch, yaw, natural

features, etc.) in order to track the position of the surface environment representation object relative to the user device in the 2D/3D space with 6DoF. In such embodiments, the surface environment representation object generation system 210 may transition between tracking systems based on the availability of the tracked indicia to maintain consistent tracking in 6DoF.

[0068] The disruption detection module 506 monitors tracking indicia to detect disruptions. Upon the disruption detection module 506 detecting an interruption of one or more indicia, such that tracking in 6DoF becomes unreliable or impossible, the surface environment representation object generation system 210 transitions to tracking the surface environment representation object in the 2D/3D space in 3DoF in order to prevent an interruption of the display. For example, the surface environment representation object generation system 210 may transition from a first tracking sub-system 504A (or first set of tracking sub-systems among the set of tracking sub-systems) that track the object in 6DoF to a second tracking sub-system 504B among the set of tracking sub-systems (or second set of tracking sub-systems), wherein the second tracking system is capable of tracking the surface environment representation object with 3DoF in the 3D space, based on the tracking indicia available.

[0069] In some example embodiments, the set of tracking systems of the surface environment representation object generation system 210 comprises a gyroscopic tracking system, an NFT system, and a SLAM tracking system. Each tracking system among the set of tracking systems may analyze tracking indicia to track a position of a virtual object within a 3D space. For example, to track a virtual object with 6DoF, the surface environment representation object generation system 210 may require at least six tracking indicia to be available. As tracking indicia become obstructed or unavailable for various reasons, the surface environment representation object generation system 210 may transition between the available tracking systems among the set of tracking systems in order to maintain 6DoF, or transition to 3DoF if necessary.

[0070] It will be readily appreciated that the surface environment representation object generation system 210 provides consistent rendered surface environment representation objects (e.g., 3D captions, avatars, or 3D animations, videos or images) in real-world 3D spaces in a wide variety of environments and situations. In many applications it can be desirable to provide firm consistency for the locations of these virtual objects as one or more users, cameras, or other tracking items move around in the environment. This can involve the recognition and use of a specific fixed reference point (e.g., a fixed surface) in the real-world environment. Not using a fixed reference point or item can result in floating or other undesirable inconsistencies in the rendering and presentation of the virtual objects.

[0071] To ensure firm consistency in the location of virtual objects, annotation data in the example form of a presentation lens that is specific for the surface environment representation object tracking and rendering described herein may be employed. In particular, a surface aware lens is a presentation lens that identifies and references a real-world surface (e.g., the ground) for the consistent rendering and presentation of surface environment representation objects in 3D space. The surface aware lens can be a specific portion or submodule within the rendering module 502. This surface aware lens of the rendering module 502 can be configured to recognize a reference surface based on visual camera content, and may also utilize other device inputs (e.g., gyroscope, accelerometer, compass) to determine what is an appropriate surface within a 3D space depicted in a live camera feed. Once the reference surface has been determined, then virtual surface environment representation object can be positioned with respect to that reference surface. In an example, the reference surface in the 3D space is a ground surface. In this example, the surface environment representation object generation system 210 renders the surface environment representation object at a position in the 3D space such that the surface environment representation object appears to be on or slightly above the 3D space. The surface environment representation object rendered by the surface environment

representation object generation system 210 includes surface textures of a first real-world environment over which the surface environment representation object was positioned in an image. In some embodiments, the surface environment representation object rendered by the surface environment representation object generation system 210 includes surface textures of another camera feed depicting other portions of the real-world environment in which the surface environment representation object was positioned in an image. Such an object is stored and used by the surface environment representation object generation system 210 to track and render the surface environment representation object of the first real-world environment in a second real-world environment depicted in another image on the same or different user device.

[0072] FIGS. 6 and 7 are flowcharts illustrating example operations of the surface environment representation object generation system in performing methods 600 and 700 for generating a message that includes a surface environment representation object, according to example embodiments. The methods 600 and 700 may be embodied in transitory or non-transitory computer-readable instructions for execution by one or more processors such that the operations of the methods 600 and 700 may be performed in part or in whole by the functional components of the surface environment representation object generation system 210; accordingly, the methods 600 and 700 are described below by way of example with reference thereto. It shall be appreciated, however, that at least some of the operations of the methods 600 and 700 may be deployed on various other hardware configurations and the methods 600 and 700 are not intended to be limited to the surface environment representation object generation system 210. Any one of the operations described in connection with methods 600 and 700 may be performed in a different order than that shown and described or entirely omitted.

[0073] At operation 602, the annotation system 206 receives a first input to activate a surface representation object lens. The surface representation object lens may be selected from a group of lenses. For example, a list of

several lens is presented over a real-time feed from a camera and the user can tap a given lens in the list to instantly add the lens to the real-time feed. As one example, the user can tap the surface representation object lens to view a list of available surface representation object lens types or objects that the user can selected from at operation 604.

[0074] At operation 604, the surface environment representation object generation system 210 causes display of an editing interface 212 on the client device 102. The editing interface 212 enables a user to input a selection of a virtual object and modifications or edits to the virtual object that provide a basis for generating a surface environment representation object from the virtual object. In some embodiments, the virtual object that is selected by the user is a 2D or 3D object and the surface environment representation object that is generated is a 2D/3D object. To this end, the editing interface 212 may include a keyboard or other input mechanism to enable the user to input a selection of 2D/3D content (e.g., the one or more text characters, image selection, video selection, animation selection, and so forth). The 2D/3D content input by the user is displayed as a 2D/3D overlay on top of a camera feed produced by a camera of the client device 102. The camera feed presents a first real-world environment of the client device 102.

[0075] At operation 606, the surface environment representation object generation system 210 receives input selecting a first virtual object by a user of the client device using the editing interface 212. The input may adjust the position of the 2D/3D object in the first real-world environment such that the 2D/3D object overlays different portions of the first real-world environment depicted in the camera feed. For example, the user can select a virtual object that includes a portion that is configured to be in a partially-transparent state or a locked state. Specifically, the virtual object can be a caption region that adds characters one at a time as the user types in the characters of the caption. Each character may include a border and pixels within the border are configured to be at least partially transparent to reveal and show the portions of the real-world environment over which the characters are positioned. As additional characters are input and/or as the caption is moved

around in 2D/3D space in the real-time camera feed, the pixels within the borders of the characters are updated such that the surface textures in a way that is partially transparent to reveal the portions of the real-world environment over which they are positioned.

[0076] The user can input an instruction to lock the state of the virtual object such that the current pixel values representing in the partially transparent manner the portion of the real-world environment over which they are positioned are locked. This allows the user to manipulate the position and orientation of the virtual object in 2D/3D space without having the pixel values updated and without having the surface textures continue updating to represent in a partially transparent manner the underlying portions of the real-world environment. This way, the virtual object can represent with its surface textures a first portion of the real-world environment over which the object was positioned when the pixels were in a partially transparent state while being positioned over a different portion of the real-world environment depicted in the real-time camera feed.

[0077] As another example, the user can select a virtual object that includes a portion dedicated or selectable to represent images from another real-time camera feed. Specifically, the virtual object can be positioned within 2D/3D space in the real-time camera feed received from a first camera (e.g., a front-facing camera) and a portion of the surface textures of the virtual object can be modified with content received from a real-time camera feed received from a second camera (e.g., a rear-facing or front facing camera). The user can dynamically change whether the front or rear- facing camera is used to provide the content for the surface textures of the virtual object. The user can input an instruction to lock the state of the virtual object such that the current pixel values representing in the currently received content of the second real-time camera feed are locked. This allows the user to manipulate the position and orientation of the virtual object in 2D/3D space without having the pixel values updated and without having the surface textures continue updating to represent the content received from the second real-time camera feed. This way, the virtual object can represent with its surface

textures a first portion of the real-world environment over which the object was positioned when the pixels were being updated with the second camera feed while being positioned over a different portion of the real-world environment depicted in the first real-time camera feed.

[0078] At operation 610, the surface environment representation object generation system 212 detects a third input and in response to detecting the third input, the surface environment representation object generation system 212 causes display of a preview interface 214, at operation 612. The third input may, for example, include a motion-based input such as a change of orientation of the client device 102. For example, if the user is pointing the camera of the client device 102 at an upward orientation, the two-dimensional representation of the 2D content is presented in the editing interface 212. If the user changes the orientation of the camera to be facing downward, the surface environment representation object generation system 212 may toggle from displaying the editing interface 212 to displaying the preview interface 214. The preview interface 214 includes a presentation of a surface environment representation object. The surface environment representation object may be rendered at a position in a 2D/3D space captured in the camera feed that is based on a detected reference surface in the 2D/3D space such as a ground or floor surface and based on the input received at operation 606.

[0079] At operation 614, the messaging system 100 generates a message that includes one or more images with the surface environment representation object applied. In generating the message, the messaging system 100 may record a video of a user specified or predetermined length (e.g., 3 seconds) that includes one or more images from the camera feed with the surface environment representation object applied. The messaging system 100 may further apply one or more user specified filters to the recorded image(s) in generating the message. The messaging system 100 may also send the surface environment representation object to a second user to enable the second user to place the surface environment representation object in a different real-time camera feed of another environment. This

way, the surface environment representation object can be generated using the content item of a first real-world environment and be rendered for display by another user in a second real-world environment.

[0080] As shown in the method 700 of FIG. 7, the method 600 may, in some embodiments, include operations 702, 704, 706, 708, and 710. Consistent with these embodiments, the operations 702, 704, 706, 708, and 710 may be performed as part of operation 606 and/or 610 (e.g., as a sub-routine or sub-operation) where the surface environment representation object generation system 210 generates a surface environment representation object from an input virtual object and causes display of a preview interface comprising a presentation of the virtual surface environment representation object within a first or second real-world environment captured within a live or recorded first or second camera feed.

[0081] At operation 702, the rendering component 502 captures by a camera of a user device a first image depicting a first environment of the user device. For example, the rendering component 502 activates a rear-facing camera of the user device and presents in full or partial screen real-time images depicting the real-world environment.

[0082] At operation 704, the rendering component 502 overlays a first virtual object on a portion of the first image depicting the first environment. For example, the rendering component 502 receives a user selection of a virtual object configured for surface environment representation. The virtual object may be a virtual object configured to present content items from the first environment over which the first virtual object is positioned in a partially or fully transparent manner. In another embodiment, the virtual object may be a virtual object configured to display images received from a second camera feed (e.g., a front or rear facing camera) capturing content items from the first environment. In such cases, the rendering component 502 may identify the region of the virtual object dedicated to displaying the images from the second camera feed. The rendering component 502 may receive user input that manipulates the location and orientation of the virtual

object in 2D/3D space relative to the real-time images of the camera feed over which the object is positioned.

[0083] At operation 706, the rendering component 502 modifies a surface of the first virtual object using content captured by the user device. For example, in such cases where the object selected is configured to present content from the real-time camera feed in a transparent or partially transparent manner, the rendering component 502 may copy into the pixel values within the border of the virtual object the pixels values of the real-time image portions over which the virtual object is positioned. The border may be opaque or also fully transparent. The pixel values may be copied on a pixels by pixel basis. Specially, the rendering component 502 may determine the 2D/3D position of the virtual object within the real-time image. The rendering component 502 may identify a first pixel position of the virtual object within the real-time image and retrieve the pixel value of the real-time image at the same first pixel position. The rendering component 502 may copy the pixel value of the real-time image at the first pixel position into the first pixel position of the virtual object. The rendering component 502 may continue processing each pixel of the virtual object in the same manner until all the pixel values are updated with the corresponding pixel values over which they are positioned of the real-time images.

[0084] In such cases where the virtual object is configured to present content from another camera feed, the rendering component 502 activates the second camera feed and receives and displays images from the second camera feed on the specified portion of the virtual object. Once the rendering component 502 receives a user request to lock the images displayed in the specified portion, the rendering component 502 locks the object and allows the user to manipulate the size, shape, and position of the object that now includes the image last received from the second camera feed prior to the request to lock. The manipulation can be performed while the virtual object is presented on the real-time camera feed received initially.

[0085] At operation 708, the rendering component 502 stores a second virtual object comprising the first virtual object with the modified surface. For example, in response to receiving the user request to lock the virtual object, the rendering component 502 maintains in the virtual object the image depicted in a transparent manner the real-world image portion over which the object was positioned just prior to receiving the request to lock the virtual object. As another example, the rendering component 502 locks the object that includes the image last received from the second camera feed prior to the request to lock. The locked virtual object is stored as the second virtual object (e.g., the surface environment representation object).

[0086] At operation 710, the rendering component 502 generates for display the second virtual object on a portion of a second image depicting a second environment. For example, the rendering component 502 allows the user to send the second virtual object to a second user for display in a real-time camera feed of the user device of the second user. As another example, the rendering component 502 allows the user to reposition and manipulate the object in the camera feed showing other portions of the first environment.

[0087] FIGS. 8-12 are interface diagrams that illustrate various interfaces provided by the messaging system, according to some example embodiments. FIG. 8 specifically shows an example of a virtual surface environment representation object being generated within a given environment of a first user device and is moved after being locked to be positioned in a second portion of the given environment of the first user device.

[0088] As shown in FIG. 8, a user activates a lens feature of the messaging application allowing a user to add a virtual object to a real-world environment depicted in the real-time images. As an example, the virtual object selected by the user is a 2D caption. In response, the rendering component 502 allows the user to input characters of the 2D caption and as the characters are input the rendering component 502 presents the characters 810 on top of the real-world images received from the real-time camera feed.

The rendering component 502 updates surface textures of the characters such that portions of the characters 810 within a border of the caption are at least partially or fully transparent to reveal the underlying real-world images over which the characters are positioned. Namely, the rendering component 502 copies pixel values from the positions of the real-world image portions over which the characters 810 are positioned to the pixels within the border of the characters.

[0089] As shown, when characters 810 are positioned over an oak colored desk, the pixels of the characters 810 within the border are updated with the same pixel values of the oak colored desk. As the camera is moved (e.g., is panned right) to show another portion of the real-world environment 830 (e.g., a person working at a desk), the pixels of the characters 810 are updated to show in a transparent manner the portions of the person working at the desk over which the characters are positioned. Characters 820 show the updated pixel values showing that characters 820 are transparent, at least portions within a border of the caption, to reveal the underlying real-world image portions. The rendering component 502 receives a user input to lock the virtual object and create a 3D surface environment representation object, such as to lock the pixel values of the caption to prevent the pixels within the border of the characters from being updated as the virtual object is moved around or as the camera feed changes. Specifically, characters 840 show the locked pixel values and are positioned in 3D space in another portion of the real-world environment (e.g., when the camera is panned left). For example, the characters 840 show the pixel values of the real-world environment 830 (e.g., a person working at a desk, when the camera was panned right) while being positioned in another portion of the real-world environment (e.g., after being panned left) in which the real-world environment 830 is no longer visible.

[0090] FIG. 9 specifically shows an example of a virtual surface environment representation object being generated within a first environment of a first user device and is sent to a second user device to be positioned and displayed after being locked in a second environment of the

second user device. As an example, the virtual object selected by the user is a 2D caption 920. In response, the rendering component 502 allows the user to input characters of the 2D caption 920 and as the characters are input the rendering component 502 presents the characters 920 on top of the real-world images 910 received from the real-time camera feed of the first user device. The rendering component 502 updates surface textures of the characters such that portions of the characters 920 within a border of the caption are at least partially or fully transparent to reveal the underlying real-world images 910 over which the characters are positioned. Namely, the rendering component 502 copies pixel values from the positions of the real-world image portions 910 over which the characters 920 are positioned to the pixels within the border of the characters 920.

[0091] The rendering component 502 receives a user input to lock the virtual object and create a 2D surface environment representation object, such as to lock the pixel values of the 2D caption to prevent the pixels within the border of the characters 920 from being updated as the virtual object is moved around, sent to a second user, or as the camera feed changes. Specifically, characters 940 show the locked pixel values and are positioned in 3D space in a second real-world environment 930 after being sent to the second user. For example, the characters 940 show the pixel values of the real-world environment 910 while being positioned in a second real-world environment 930 of a second user device in which the real-world environment 910 is no longer visible.

[0092] FIG. 10 specifically shows an example of a virtual surface environment representation object being generated within a given environment of a first user device and is re-oriented in 3D space after being locked within the same given environment of the first user device. As shown in FIG. 10, a user activates a lens feature of the messaging application allowing a user to add a virtual object to a real-world environment depicted in the real-time images. As an example, the virtual object selected by the user is a 2D caption. In response, the rendering component 502 allows the user to input characters of the 2D caption 1020

and as the characters are input the rendering component 502 presents the characters 1020 on top of the real-world images 1010 received from the real-time camera feed. The rendering component 502 updates surface textures of the characters 1020 such that portions of the characters 1020 within a border of the caption are at least partially or fully transparent to reveal the underlying real-world images 1010 over which the characters are positioned. Namely, the rendering component 502 copies pixel values from the 2D and 3D positions of the real-world image 1010 portions over which the characters 1020 are positioned to the pixels within the border of the characters.

[0093] The rendering component 502 receives a user input to lock the virtual object and create a 3D surface environment representation object, such as to lock the pixel values of the caption to prevent the pixels within the border of the characters from being updated as the virtual object is moved around or as the camera feed changes. In response, the rendering component 502 creates a 3D surface environment representation object 1030 that maintains the current values of a first surface 1032 (e.g., a front surface) of the 3D surface environment representation object (to represent the underlying real-world image portion 1010) and a second surface 1034 (e.g., a perpendicular surface to the first surface in 3D space) is replaced with a single pixel value. This makes 3D surface environment representation object 1030 appear to be coming out of the screen and positioned and oriented in 3D space. The 3D surface environment representation object 1030 can be reoriented in 3D space to appear to lay down on a road as shown by 3D surface environment representation object 1040 being re-oriented in 3D space of the same real-world images 1010. A first surface 1032 of the 3D surface environment representation object 1040 has pixel values of the real-world images 1010 portions over which the object was positioned before being locked and re-oriented over a different portion of the real-world images 1010. As shown, a portion 1042 of the real-world images 1010 (over which a portion of the virtual object was positioned before being locked) is shown in a portion 1044 of the 3D surface environment representation object

1040 (after being locked and re-oriented) even though the 3D surface environment representation object 1040 no longer is positioned on top of the portion 1042.

[0094] FIG. 11 specifically shows an example 1100 of a virtual surface environment representation object being generated within a given environment of a first user device and is re-oriented in 3D space after being locked within the same given environment of the first user device. As shown in FIG. 11, a user activates a lens feature of the messaging application allowing a user to add a virtual object 1120 to a real-world environment depicted in the real-time images 1110. As an example, the virtual object selected by the user is a virtual hat 1120. In response, the rendering component 502 allows the user to position the virtual hat in 2D/3D space on top of the real-world images 1110 (e.g., a person's head) received from a first real-time camera feed. The rendering component 502 identifies a portion 1130 of the virtual object 1120 which is configured to present content from a second real-time camera feed. The rendering component 502 activates a second camera (e.g., selected by the user as the front or rear facing camera), receives real-time images from the second camera and presents those images from the second camera on the portion 1130 of the virtual object 1120. In this way, multiple camera feeds are shown in example 1100, such that the virtual object 1120 is positioned over images from a first camera feed and images from a second camera feed are placed on a portion of the virtual object. In some embodiments, rather than activating a second camera feed, the rendering component 502 allows a user to select an image, video or animation, to insert into the portion 1130. In such cases, one camera feed is utilized, such that the virtual object 1120 is positioned over images from the camera feed and an image, video or animation selected by the user is placed on a dedicated or user-selectable portion 1130 of the virtual object 1120.

[0095] As shown in FIG. 12A, the rendering component 502 receives a user input to lock the virtual object 1120, such as to lock the pixel values of the virtual object 1120 to prevent the pixels from being updated with the second

camera feed or with the selected image, video or animation as the virtual object 1120 is moved around or as the first camera feed changes. In response, the rendering component 502 creates a surface environment representation object 1210 that maintains the current values of a portion 1230 (e.g., a front surface of the virtual hat) as the surface environment representation object 1210 is moved around or as the camera feed changes. Specifically, portion 1230 shows the locked pixel values of the surface environment representation object 1210 and do not change as the surface environment representation object 1210 is positioned in 3D space in another portion of the real-world environment 1220 (e.g., when the camera is panned left).

[0096] In some embodiments, the virtual object 1120 displayed on the first user device is sent in a message to a second user device. FIG. 12B shows the virtual object 1120, created by the first user device, being received by a second user device. The second user device may be operated by a second user and may present real-time images from a camera feed of the second user device. In FIG. 12B, the real-time camera feed presents a face 1284 of the second user. The virtual object 1120, which may be a virtual hat, may be positioned as virtual object 1282 on top of the head of the second user, as shown in FIG. 12B. Any image, video or animation, that was inserted by the first user device into the portion of the virtual object is also presented in the portion 1280 of the virtual object 1282. In this way, a first user may create a virtual object which is presented on a first user device, as shown in FIG. 11. The first user device may present the virtual object as being worn by the first user. The first user may send the created virtual object to a second user. As shown in FIG. 12B, a second user device of second user may present the received virtual object to the second user and may position the virtual object to make it appear as though the virtual object is also being worn by the second user.

[0097] FIG. 13 is a block diagram illustrating an example software architecture 1306, which may be used in conjunction with various hardware architectures herein described. FIG. 13 is a non-limiting example of a

software architecture and it will be appreciated that many other architectures may be implemented to facilitate the functionality described herein. The software architecture 1306 may execute on hardware such as machine 1300 of FIG. 13 that includes, among other things, processors 1304, memory 1314, and input/output (I/O) components 1319. A representative hardware layer 1352 is illustrated and can represent, for example, the machine 1300 of FIG. 13. The representative hardware layer 1352 includes a processing unit 1354 having associated executable instructions 1304. Executable instructions 1304 represent the executable instructions of the software architecture 1306, including implementation of the methods, components and so forth described herein. The hardware layer 1352 also includes memory and/or storage modules memory/storage 1356, which also have executable instructions 1304. The hardware layer 1352 may also comprise other hardware 1358.

[0098] In the example architecture of FIG. 13, the software architecture 1306 may be conceptualized as a stack of layers where each layer provides particular functionality. For example, the software architecture 1306 may include layers such as an operating system 1302, libraries 1320, applications 1316, frameworks/middleware 1318, and a presentation layer 1314. Operationally, the applications 1316 and/or other components within the layers may invoke API calls 1308 through the software stack and receive a response 1312 as in response to the API calls 1308. The layers illustrated are representative in nature and not all software architectures have all layers. For example, some mobile or special purpose operating systems may not provide a frameworks/middleware 1318, while others may provide such a layer. Other software architectures may include additional or different layers.

[0099] The operating system 1302 may manage hardware resources and provide common services. The operating system 1302 may include, for example, a kernel 1322, services 1324, and drivers 1326. The kernel 1322 may act as an abstraction layer between the hardware and the other software layers. For example, the kernel 1322 may be responsible for memory management, processor management (e.g., scheduling), component management, networking, security settings, and so on. The services 1324

may provide other common services for the other software layers. The drivers 1326 are responsible for controlling or interfacing with the underlying hardware. For instance, the drivers 1326 include display drivers, camera drivers, Bluetooth® drivers, flash memory drivers, serial communication drivers (e.g., Universal Serial Bus (USB) drivers), Wi-Fi® drivers, audio drivers, power management drivers, and so forth depending on the hardware configuration.

[00100] The libraries 1320 provide a common infrastructure that is used by the applications 1316 and/or other components and/or layers. The libraries 1320 provide functionality that allows other software components to perform tasks in an easier fashion than to interface directly with the underlying operating system 1302 functionality (e.g., kernel 1322, services 1324 and/or drivers 1326). The libraries 1320 may include system libraries 1344 (e.g., C standard library) that may provide functions such as memory allocation functions, string manipulation functions, mathematical functions, and the like. In addition, the libraries 1320 may include API libraries 1346 such as media libraries (e.g., libraries to support presentation and manipulation of various media format such as MPREG4, H.264, MP3, AAC, AMR, JPG, PNG), graphics libraries (e.g., an OpenGL framework that may be used to render two-dimensional and 3D in a graphic content on a display), database libraries (e.g., SQLite that may provide various relational database functions), web libraries (e.g., WebKit that may provide web browsing functionality), and the like. The libraries 1320 may also include a wide variety of other libraries 1348 to provide many other APIs to the applications 1316 and other software components/modules.

[00101] The frameworks/middleware 1318 (also sometimes referred to as middleware) provide a higher-level common infrastructure that may be used by the applications 1316 and/or other software components/modules. For example, the frameworks/middleware 1318 may provide various graphic user interface (GUI) functions, high-level resource management, high-level location services, and so forth. The frameworks/middleware 1318 may provide a broad spectrum of other APIs that may be utilized by the

applications 1316 and/or other software components/modules, some of which may be specific to a particular operating system 1302 or platform.

[00102] The applications 1316 include built-in applications 1338 and/or third-party applications 1340. Examples of representative built-in applications 1338 may include, but are not limited to, a contacts application, a browser application, a book reader application, a location application, a media application, a messaging application, and/or a game application. Third-party applications 1340 may include an application developed using the ANDROID™ or IOS™ software development kit (SDK) by an entity other than the vendor of the particular platform, and may be mobile software running on a mobile operating system such as IOS™, ANDROID™, WINDOWS® Phone, or other mobile operating systems. The third-party applications 1340 may invoke the API calls 1308 provided by the mobile operating system (such as operating system 1302) to facilitate functionality described herein.

[00103] The applications 1316 may use built in operating system functions (e.g., kernel 1322, services 1324, and/or drivers 1326), libraries 1320, and frameworks/middleware 1318 to create user interfaces to interact with users of the system. Alternatively, or additionally, in some systems interactions with a user may occur through a presentation layer, such as presentation layer 1314. In these systems, the application/component "logic" can be separated from the aspects of the application/component that interact with a user.

[00104] FIG. 14 is a block diagram illustrating components of a machine 1400, according to some example embodiments, able to read instructions from a machine-readable medium (e.g., a machine-readable storage medium) and perform any one or more of the methodologies discussed herein. Specifically, FIG. 14 shows a diagrammatic representation of the machine 1400 in the example form of a computer system, within which instructions 1410 (e.g., software, a program, an application, an applet, an app, or other executable code) for causing the machine 1400 to perform any one or more of the methodologies discussed herein may be executed. As such, the

instructions 1410 may be used to implement modules or components described herein. The instructions 1410 transform the general, non-programmed machine 1400 into a particular machine 1400 programmed to carry out the described and illustrated functions in the manner described. In alternative embodiments, the machine 1400 operates as a standalone device or may be coupled (e.g., networked) to other machines. In a networked deployment, the machine 1400 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 1400 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 smart phone, a mobile device, a wearable device (e.g., a smart watch), 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 1410, sequentially or otherwise, that specify actions to be taken by machine 1400. Further, while only a single machine 1400 is illustrated, the term "machine" shall also be taken to include a collection of machines that individually or jointly execute the instructions 1410 to perform any one or more of the methodologies discussed herein.

[00105] The machine 1400 may include processors 1404, memory memory/storage 1406, and I/O components 1418, which may be configured to communicate with each other such as via a bus 1402. In an example embodiment, the processors 1404 (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 1408 and a processor 1412 that may execute the instructions 1410. The term "processor" is intended to include multi-core processors 1404 that may

comprise two or more independent processors (sometimes referred to as "cores") that may execute instructions contemporaneously. Although FIG. 14 shows multiple processors, the machine 1400 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 multiple cores, or any combination thereof.

[00106] The memory/storage 1406 may include a memory 1414, such as a main memory, or other memory storage, and a storage unit 1416, both accessible to the processors 1404 such as via the bus 1402. The storage unit 1416 and memory 1414 store the instructions 1410 embodying any one or more of the methodologies or functions described herein. The instructions 1410 may also reside, completely or partially, within the memory 1414, within the storage unit 1416, within at least one of the processors 1404 (e.g., within the processor's cache memory), or any suitable combination thereof, during execution thereof by the machine 1400. Accordingly, the memory 1414, the storage unit 1416, and the memory of processors 1404 are examples of machine-readable media.

[00107] The I/O components 1418 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 1418 that are included in a particular machine 1400 will depend on the type of machine. For example, portable machines such as mobile phones will likely 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 1418 may include many other components that are not shown in FIG. 14. The I/O components 1418 are grouped according to functionality merely for simplifying the following discussion and the grouping is in no way limiting. In various example embodiments, the I/O components 1418 may include output components 1426 and input components 1428. The output components 1426 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 input components 1428 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 other pointing instrument), tactile input components (e.g., a physical button, a touch screen that provides location and/or force of touches or touch gestures, or other tactile input components), audio input components (e.g., a microphone), and the like.

[00108] In further example embodiments, the I/O components 1418 may include biometric components 1430, motion components 1434, environmental components 1436, or position components 1438 among a wide array of other components. For example, the biometric components 1430 may 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 1434 may include acceleration sensor components (e.g., accelerometer), gravitation sensor components, rotation sensor components (e.g., gyroscope), and so forth. The environment components 1436 may include, for example, illumination sensor components (e.g., photometer), temperature sensor components (e.g., one or more thermometer 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. The position components 1438 may 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.

[00109] Communication may be implemented using a wide variety of technologies. The I/O components 1418 may include communication components 1440 operable to couple the machine 1400 to a network 1432 or devices 1420 via coupling 1424 and coupling 1422, respectively. For example, the communication components 1440 may include a network interface component or other suitable device to interface with the network 1432. In further examples, communication components 1440 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 1420 may be another machine or any of a wide variety of peripheral devices (e.g., a peripheral device coupled via a USB).

[00110] Moreover, the communication components 1440 may detect identifiers or include components operable to detect identifiers. For example, the communication components 1440 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 1440, such as, location via Internet Protocol (IP) geo-location, location via Wi-Fi®

signal triangulation, location via detecting a NFC beacon signal that may indicate a particular location, and so forth.

Glossary:

[00111] "CARRIER SIGNAL" in this context refers to any intangible medium that is capable of storing, encoding, or carrying 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 transmission medium via a network interface device and using any one of a number of well-known transfer protocols.

[00112] "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.

[00113] "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 (1xRTT), 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.

[00114] "EMPHEMERAL 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.

[00115] "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 be 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.

[00116] "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 embodiments, 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. 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 Application Specific Integrated Circuit (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 embodiments 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. 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 embodiments 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. 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 embodiments, 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 example embodiments, the processors or processor-implemented components may be distributed across a number of geographic locations.

[00117] "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.

[00118] "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.

What is claimed is:

1. A method comprising:

capturing, by a camera of a user device, a first image depicting a first environment of the user device;

overlaying, by one or more processors, a first virtual object on a portion of the first image depicting the first environment;

modifying, by the one or more processors, a surface of the first virtual object using content captured by the user device;

storing, by the one or more processors, a second virtual object comprising the first virtual object with the modified surface; and

generating for display, by the one or more processors, the second virtual object on a portion of a second image depicting a second environment.

2. The method of claim 1, wherein the first virtual object comprises a caption with text, and wherein the content captured by the user device comprises the portion of the first image such that the surface of the first virtual object is modified to represent visual components of the first environment depicted in the portion of the first image.

3. The method of claim 2, wherein modifying the surface comprises:

identifying a first pixel of the portion of the image on which a surface pixel the surface of the first virtual object is positioned;

retrieving a pixel value of the first pixel; and

copying the retrieved pixel value to the surface pixel of the first virtual caption.

4. The method of claim 3 further comprising:

identifying an outline of the caption, wherein the surface of the caption is bounded by the outline of the caption; and

selecting the surface pixel from a plurality of surface pixels within the outline of the caption.

5. The method of claim 2, wherein the second virtual object is displayed within a camera feed captured using a messaging application of the user device of the second environment, and wherein the caption comprises a three-dimensional caption.

6. The method of claim 2, wherein the portion of the first image is a first portion of the first image, further comprising:

receiving user input that moves the caption from overlaying the first portion of the first image to overlying a second portion of the first image; and

modifying the surface of the caption to represent visual components of the first environment depicted in the second portion of the first image.

7. The method of claim 2, wherein the graphical object is stored in response to receiving user input confirming a position of the caption in the first image.

8. The method of claim 1, wherein the user device is a first user device, further comprising sending the second virtual object to a second user device via a messaging application of the first user device.

9. The method of claim 8, wherein the second image depicting the second environment is captured by the second user device, further comprising generating for display the second virtual object on the portion of the second image depicting the second environment, using a messaging application of the second user device, in response to receiving the first virtual object from the first user device.

10. The method of claim 9 further comprising displaying visual components of the first environment depicted in the portion of the first image within the second environment depicted in the second image using the second virtual object.

11. The method of claim 1, wherein the first image is a first of a plurality of images of a video depicting surroundings of the user device, and wherein the second image is a second of the plurality of images of the video depicting the surroundings of the user device.

12. The method of claim 1 further comprising:

receiving user input comprising a plurality of characters of a string; and

generating first virtual object comprising the string as each character in the string is received in the user input.

13. The method of claim 1, wherein the first image is a first of a plurality of images of a first video captured by the camera of the user device, wherein the content captured by the user device comprises a second video captured by the camera of the user device.

14. The method of claim 13, wherein:

modifying the surface of the first virtual object comprises presenting the second video on the first virtual object; and

generating for display the second virtual object on the portion of the second image depicting the second environment comprises presenting the second virtual object on top of the plurality of images of the first video.

15. The method of claim 14, wherein the user device is a first user device, further comprising:

capturing a third video using a camera of a second user device; and

presenting the second virtual object on top of the third video on the second user device.

16. The method of claim 13, wherein the camera of the user device comprises a front-facing camera of the user device, wherein the first image is a first of a plurality of images of a first video captured by the front-facing camera

of the user device, wherein the content captured by the user device comprises a second video captured by a rear-facing camera of the user device.

17. The method of claim 1, wherein the first virtual object comprises a virtual hat, wherein the first image depicting the first environment further depicts a user of the user device, wherein the virtual hat is positioned on top of a head of the user depicted in the first image, and wherein the surface of the first virtual object is modified such that a portion of the virtual hat depicts a face of the user or surroundings of the user.

18. A system comprising:

one or more processors;

a memory storing instructions which, when executed by the one or more processors, cause the one or more processors to perform operations comprising:

capturing by a camera of a user device, a first image depicting a first environment of the user device;

overlaying a first virtual object on a portion of the first image depicting the first environment;

modifying a surface of the first virtual object using content captured by the user device;

storing a second virtual object comprising the first virtual object with the modified surface; and

generating for display the second virtual object on a portion of a second image depicting a second environment.

19. The system of claim 18, wherein the first virtual object comprises a caption with text, and wherein the content captured by the user device comprises the portion of the first image such that the surface of the first virtual object is modified to represent visual components of the first environment depicted in the portion of the first image.

20. A non-transitory machine-readable medium storing non-transitory instructions which, when executed by one or more processors of a machine, cause the machine to perform operations comprising:

capturing by a camera of a user device, a first image depicting a first environment of the user device;

overlaying a first virtual object on a portion of the first image depicting the first environment;

modifying a surface of the first virtual object using content captured by the user device;

storing a second virtual object comprising the first virtual object with the modified surface; and

generating for display the second virtual object on a portion of a second image depicting a second environment.

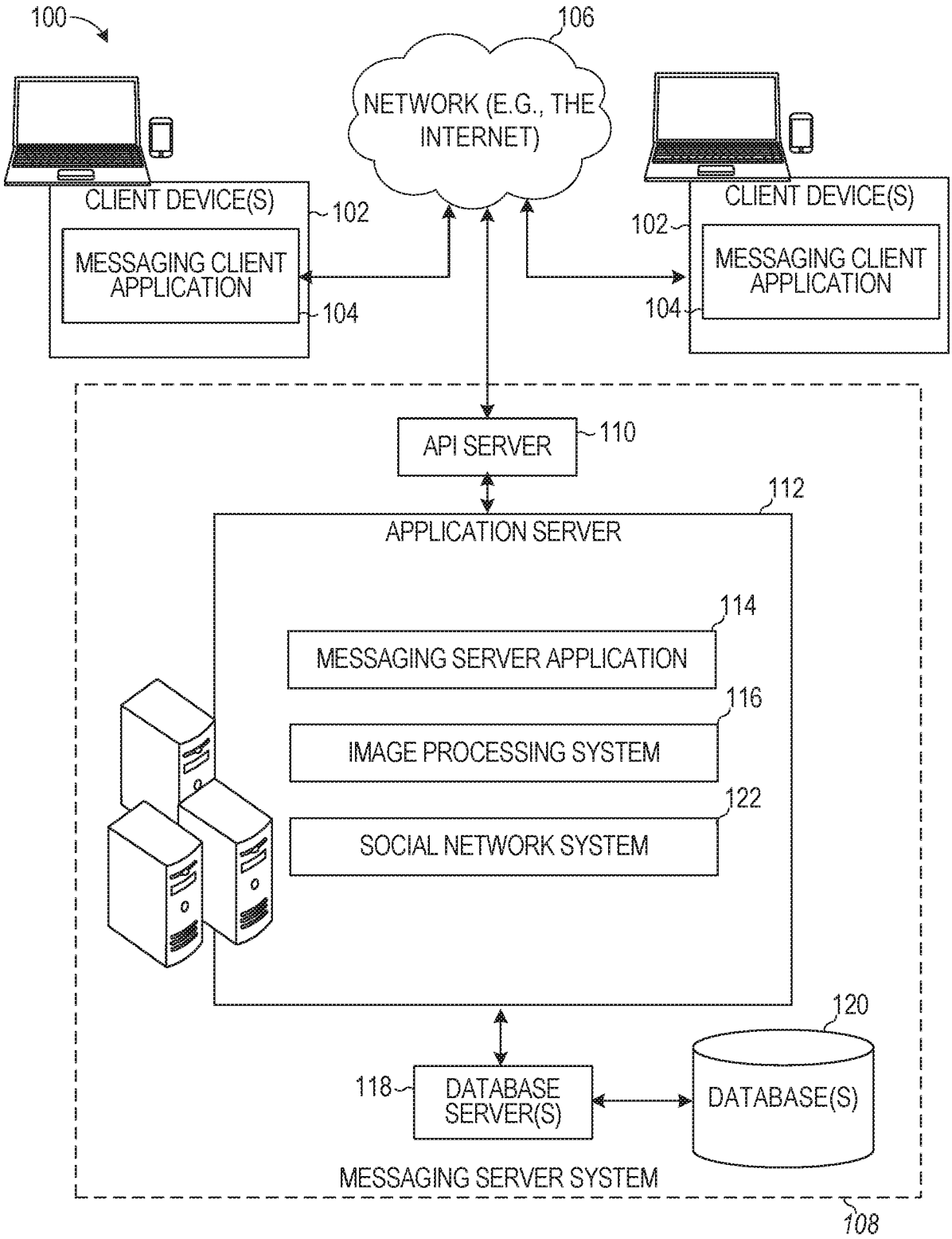

100
106
NETWORK (E.G., THE INTERNET)
CLIENT DEVICE(S)
102
MESSAGING CLIENT APPLICATION
104
CLIENT DEVICE(S)
102
MESSAGING CLIENT APPLICATION
104
API SERVER
110
112
APPLICATION SERVER
114
MESSAGING SERVER APPLICATION
116
IMAGE PROCESSING SYSTEM
122
SOCIAL NETWORK SYSTEM
120
118
DATABASE SERVER(S)
DATABASE(S)
MESSAGING SERVER SYSTEM
108

FIG. 1

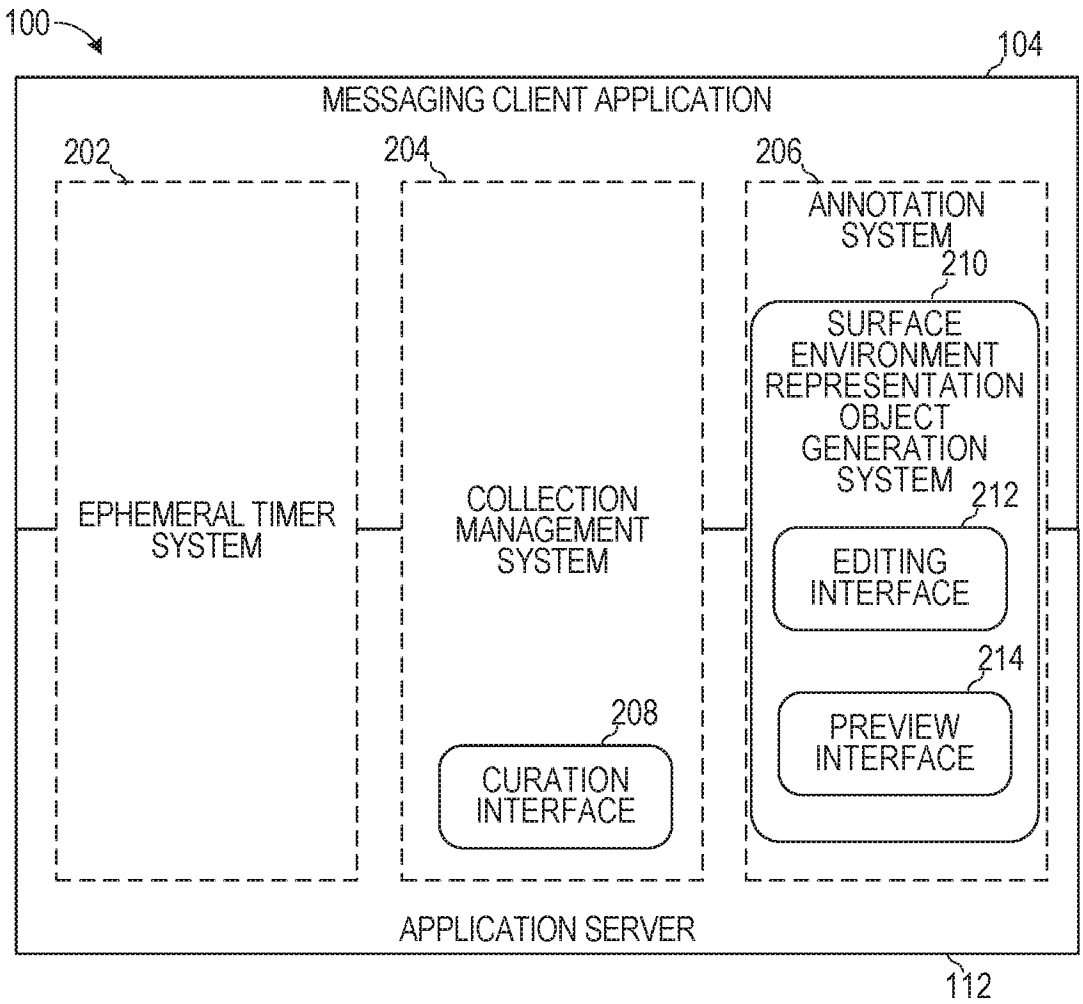

100
104
MESSAGING CLIENT APPLICATION
202
204
206
EPHEMERAL TIMER SYSTEM
COLLECTION MANAGEMENT SYSTEM
ANNOTATION SYSTEM
210
SURFACE ENVIRONMENT REPRESENTATION OBJECT GENERATION SYSTEM
212
EDITING INTERFACE
214
PREVIEW INTERFACE
208
CURATION INTERFACE
APPLICATION SERVER
112

FIG. 2

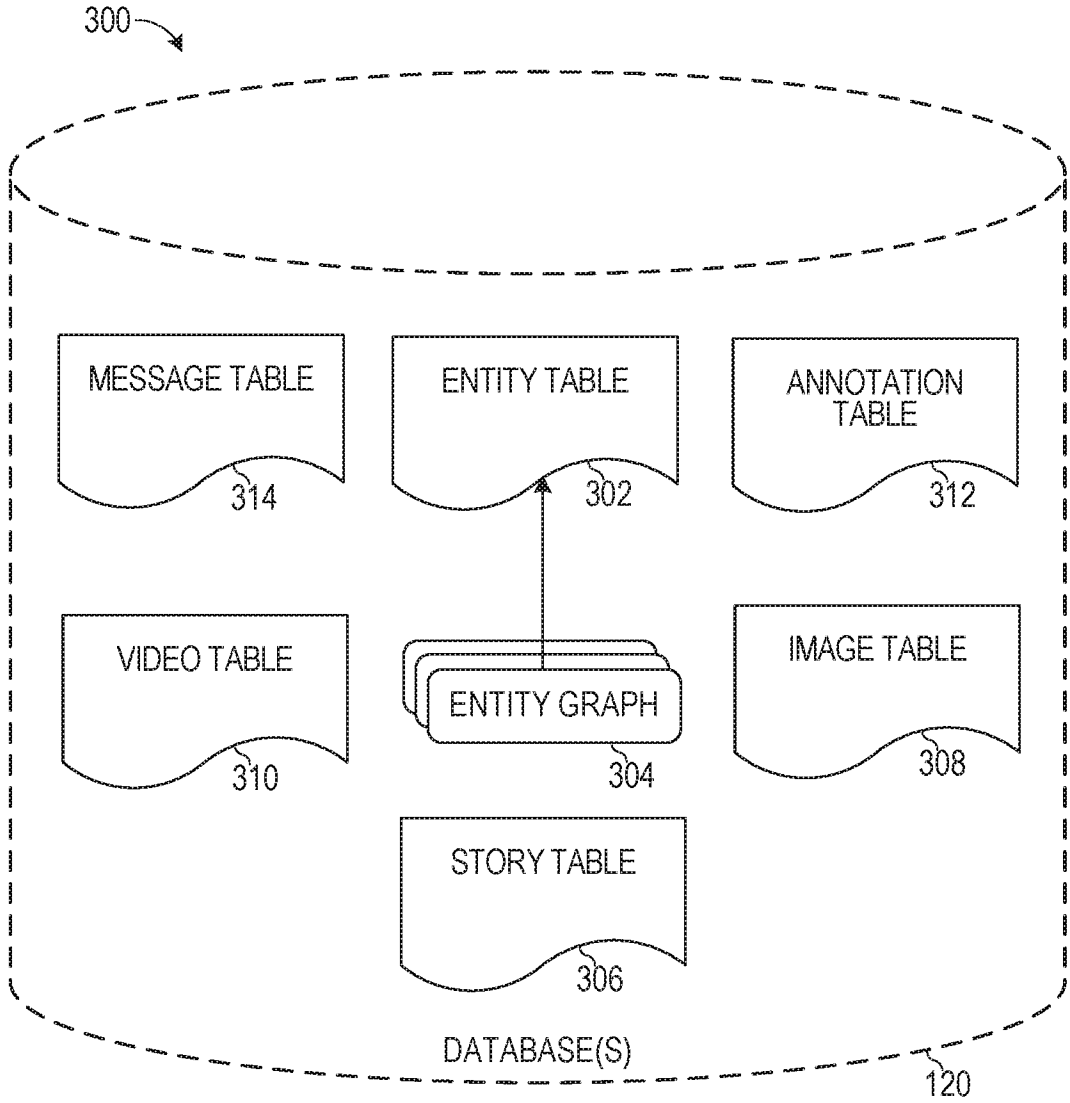

300
MESSAGE TABLE
314
ENTITY TABLE
302
ANNOTATION TABLE
312
VIDEO TABLE
310
ENTITY GRAPH
304
IMAGE TABLE
308
STORY TABLE
306
DATABASE(S)
120

FIG. 3

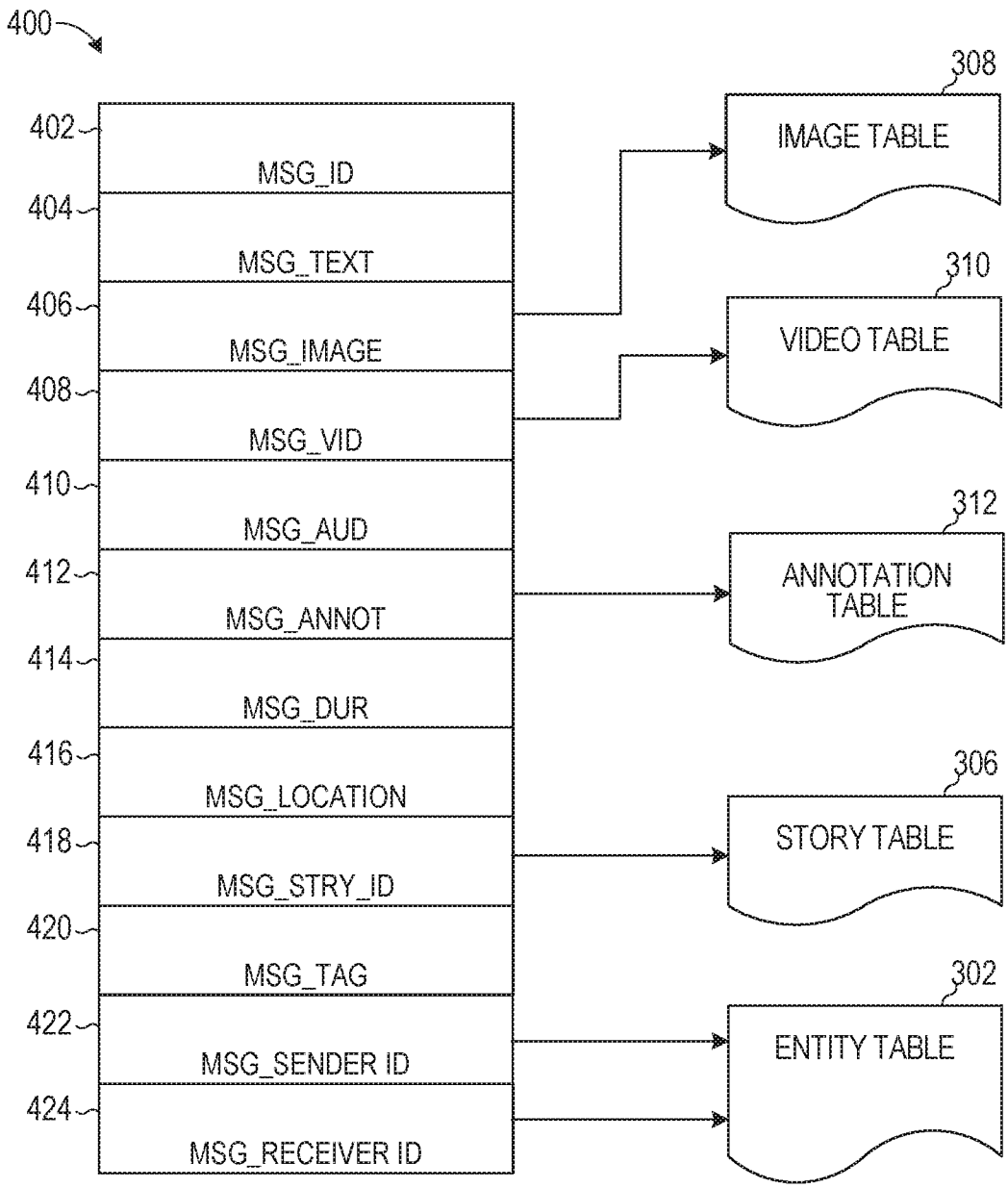

400
402
MSG_ID
404
MSG_TEXT
406
MSG_IMAGE
408
MSG_VID
410
MSG_AUD
412
MSG_ANNOT
414
MSG_DUR
416
MSG_LOCATION
418
MSG_STRY_ID
420
MSG_TAG
422
MSG_SENDER ID
424
MSG_RECEIVER ID
308
IMAGE TABLE
310
VIDEO TABLE
312
ANNOTATION TABLE
306
STORY TABLE
302
ENTITY TABLE

FIG. 4

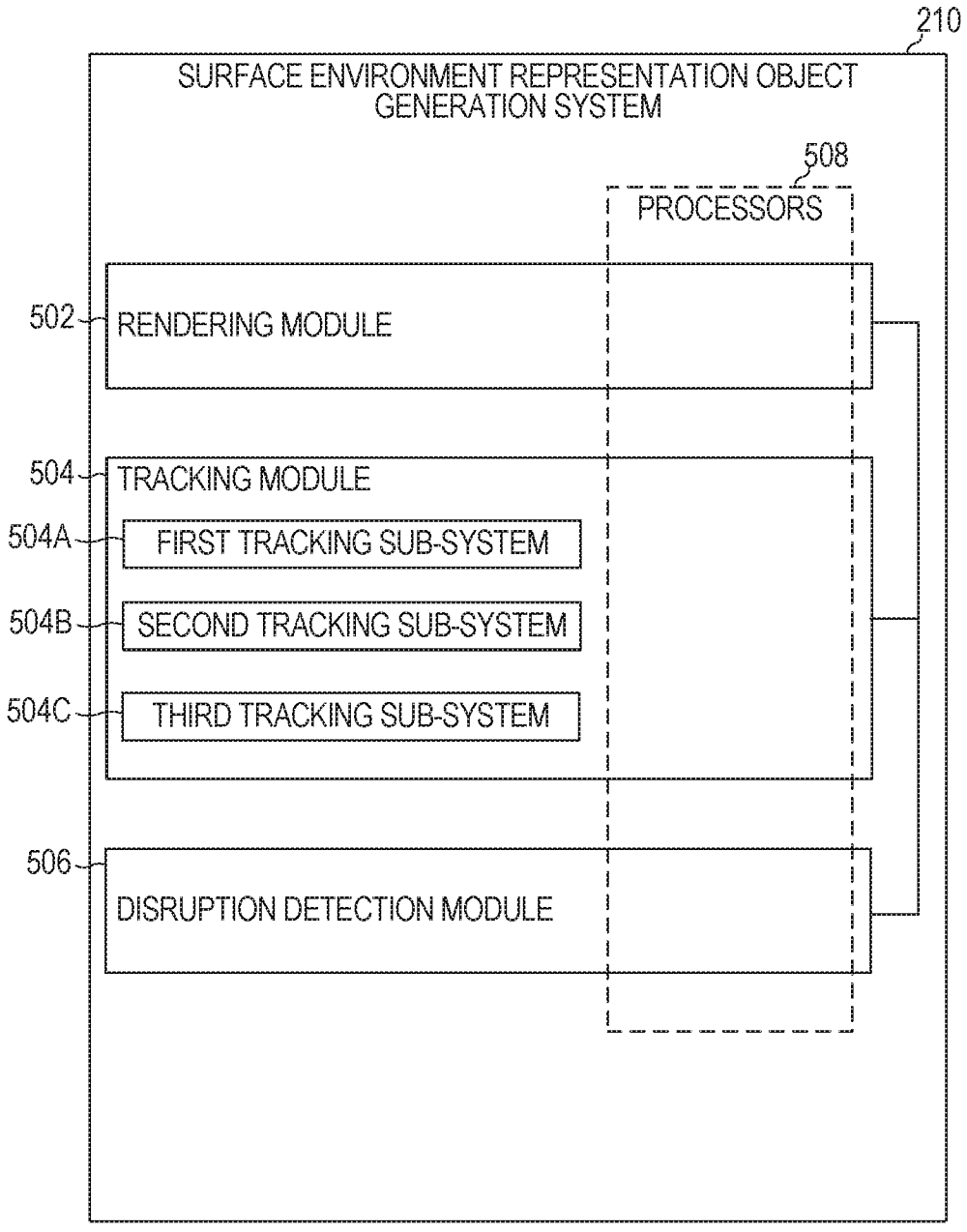

210
SURFACE ENVIRONMENT REPRESENTATION OBJECT GENERATION SYSTEM
508
PROCESSORS
502
RENDERING MODULE
504
TRACKING MODULE
504A
FIRST TRACKING SUB-SYSTEM
504B
SECOND TRACKING SUB-SYSTEM
504C
THIRD TRACKING SUB-SYSTEM
506
DISRUPTION DETECTION MODULE

FIG. 5

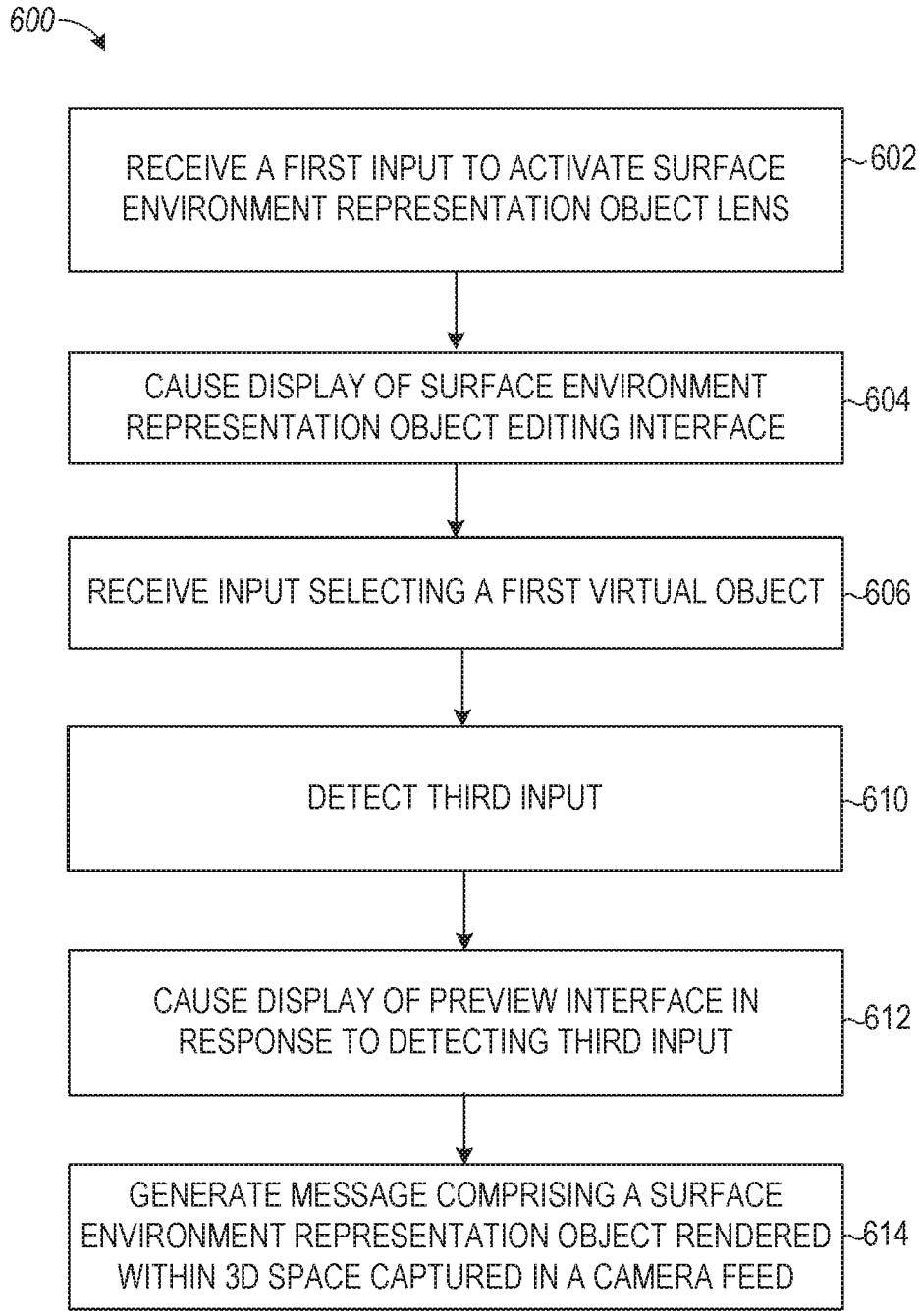

600
RECEIVE A FIRST INPUT TO ACTIVATE SURFACE ENVIRONMENT REPRESENTATION OBJECT LENS
602
CAUSE DISPLAY OF SURFACE ENVIRONMENT REPRESENTATION OBJECT EDITING INTERFACE
604
RECEIVE INPUT SELECTING A FIRST VIRTUAL OBJECT
606
DETECT THIRD INPUT
610
CAUSE DISPLAY OF PREVIEW INTERFACE IN RESPONSE TO DETECTING THIRD INPUT
612
GENERATE MESSAGE COMPRISING A SURFACE ENVIRONMENT REPRESENTATION OBJECT RENDERED WITHIN 3D SPACE CAPTURED IN A CAMERA FEED
614

FIG. 6

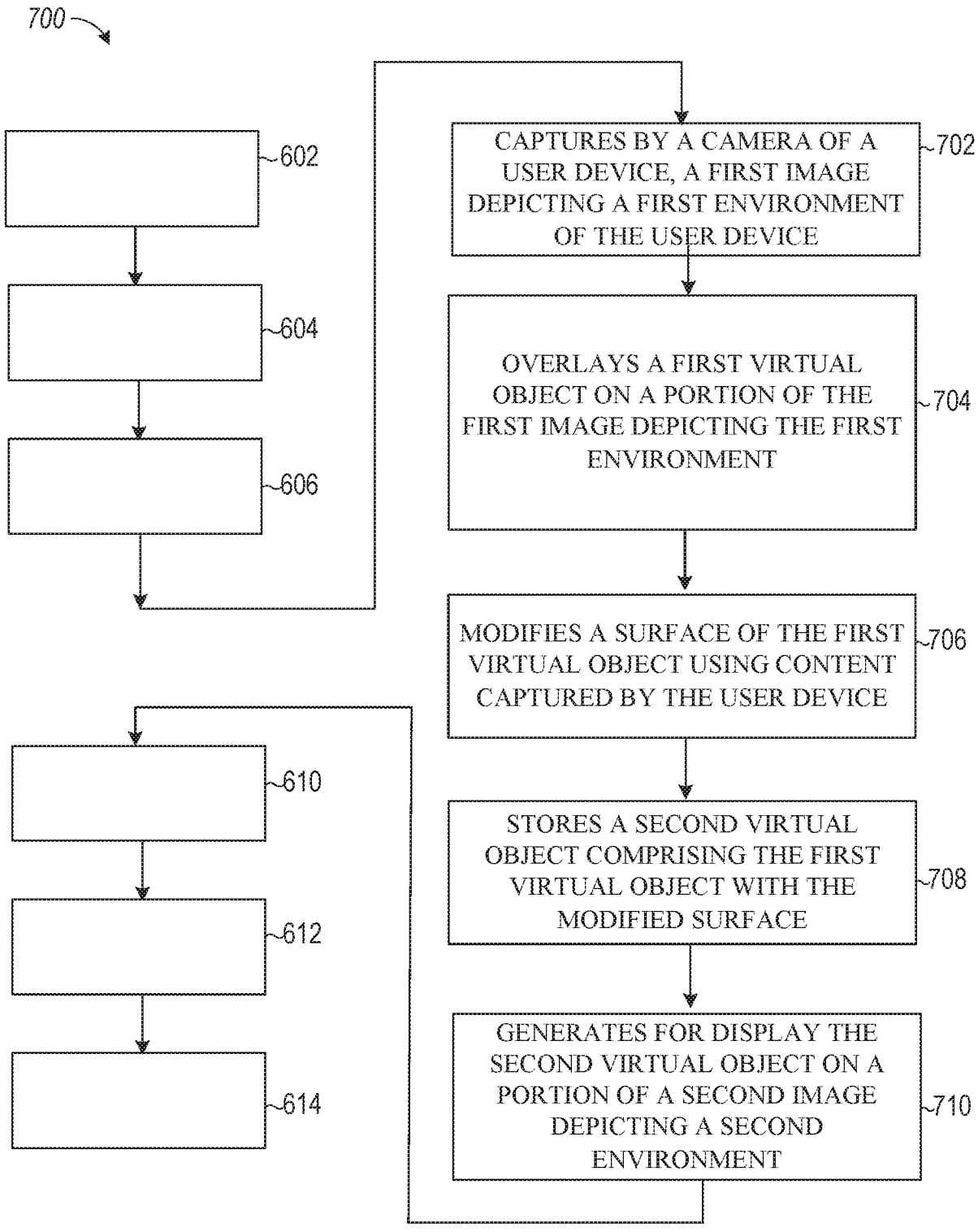

700
602
604
606
610
612
614
CAPTURES BY A CAMERA OF A USER DEVICE, A FIRST IMAGE DEPICTING A FIRST ENVIRONMENT OF THE USER DEVICE
702
OVERLAYS A FIRST VIRTUAL OBJECT ON A PORTION OF THE FIRST IMAGE DEPICTING THE FIRST ENVIRONMENT
704
MODIFIES A SURFACE OF THE FIRST VIRTUAL OBJECT USING CONTENT CAPTURED BY THE USER DEVICE
706
STORES A SECOND VIRTUAL OBJECT COMPRISING THE FIRST VIRTUAL OBJECT WITH THE MODIFIED SURFACE
708
GENERATES FOR DISPLAY THE SECOND VIRTUAL OBJECT ON A PORTION OF A SECOND IMAGE DEPICTING A SECOND ENVIRONMENT
710

FIG. 7

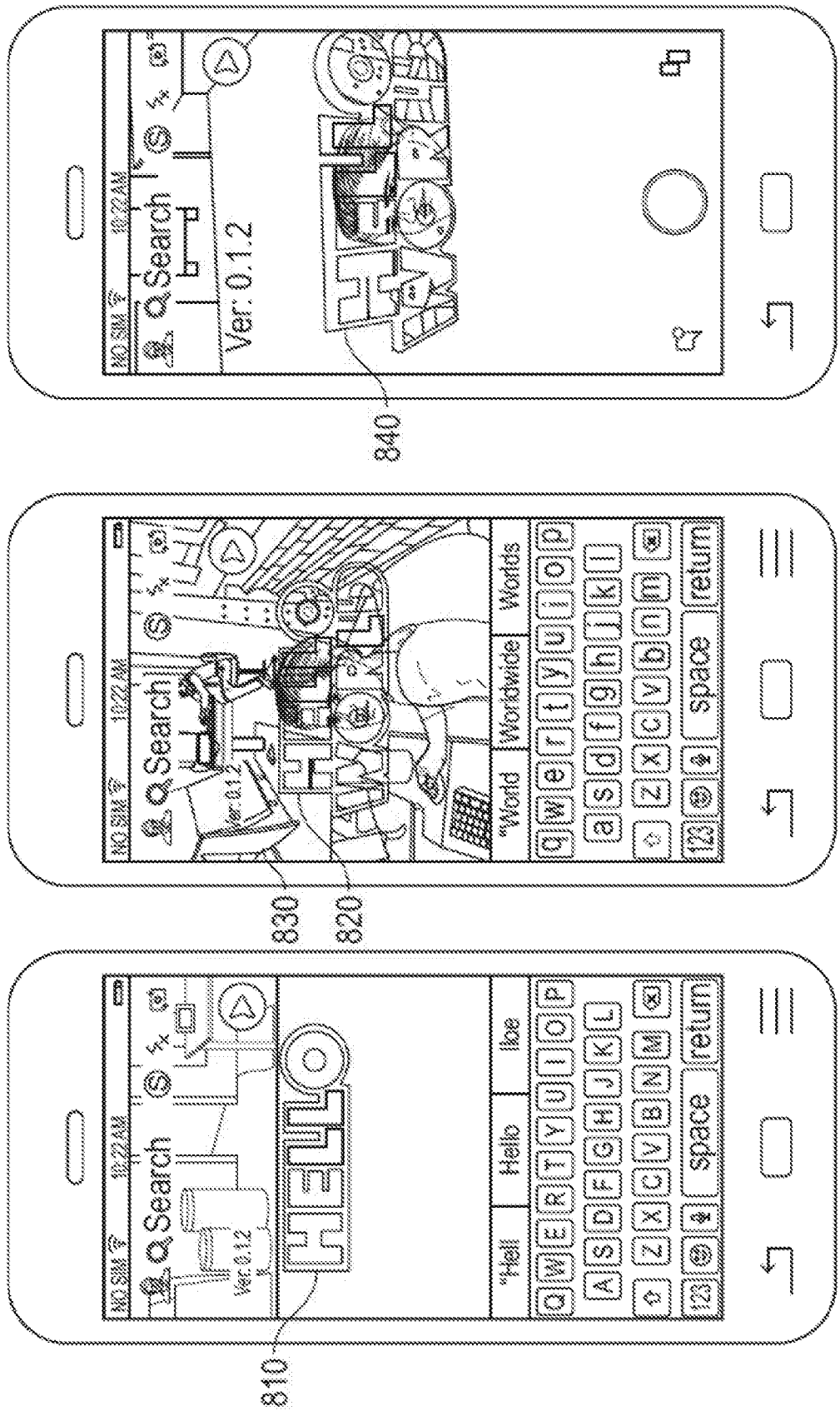

NO SIM
10:22 AM
Search
Ver: 0.1.2
HELLO
WORLD
"Hell
Hello
lloe
"World
Worldwide
Worlds
space
return
810
820
830
840

FIG. 8

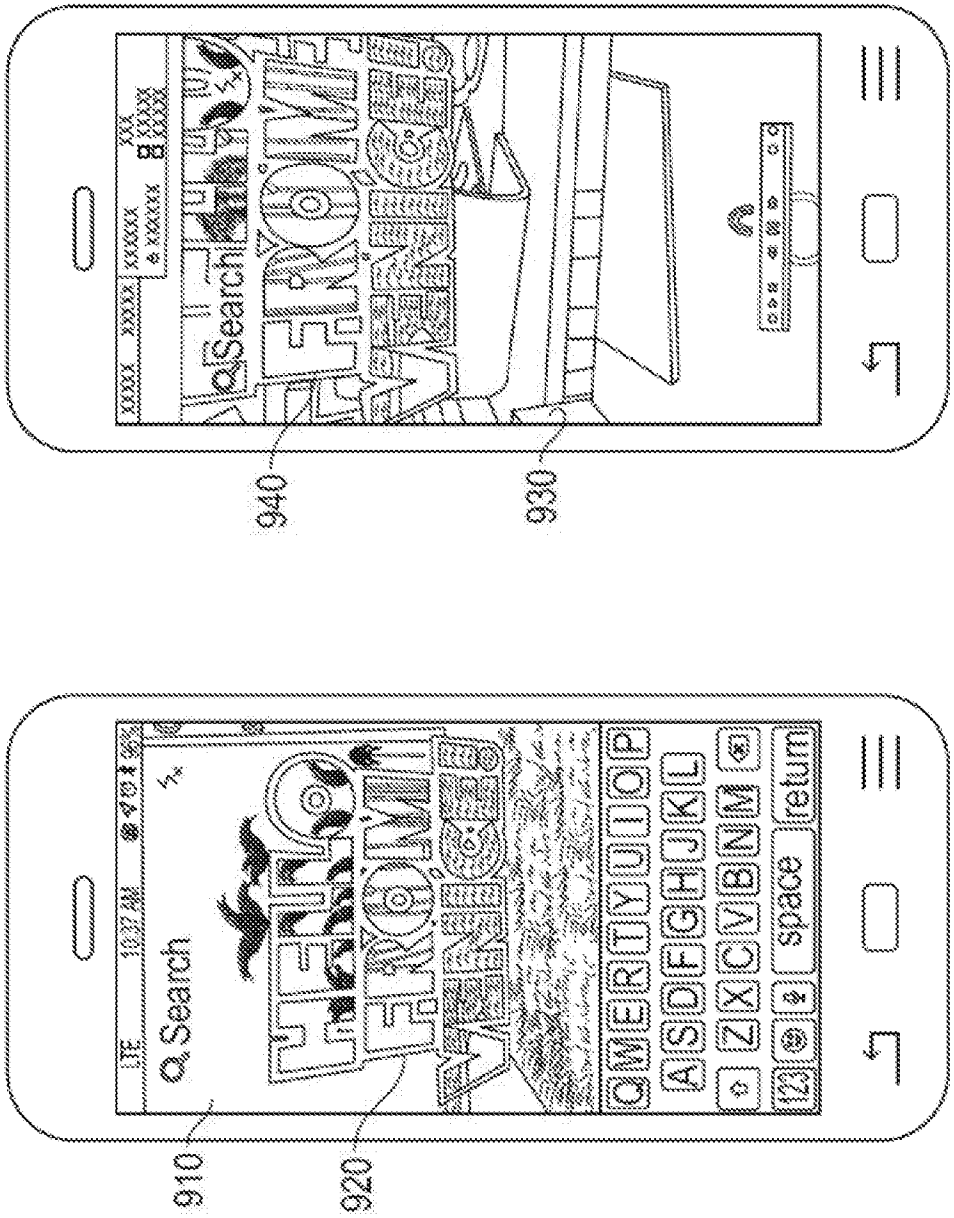

910
920
930
940
Search
FINDING
QWERTYUIOP
ASDFGHJKL
ZXCVBNM
space
return

FIG. 9

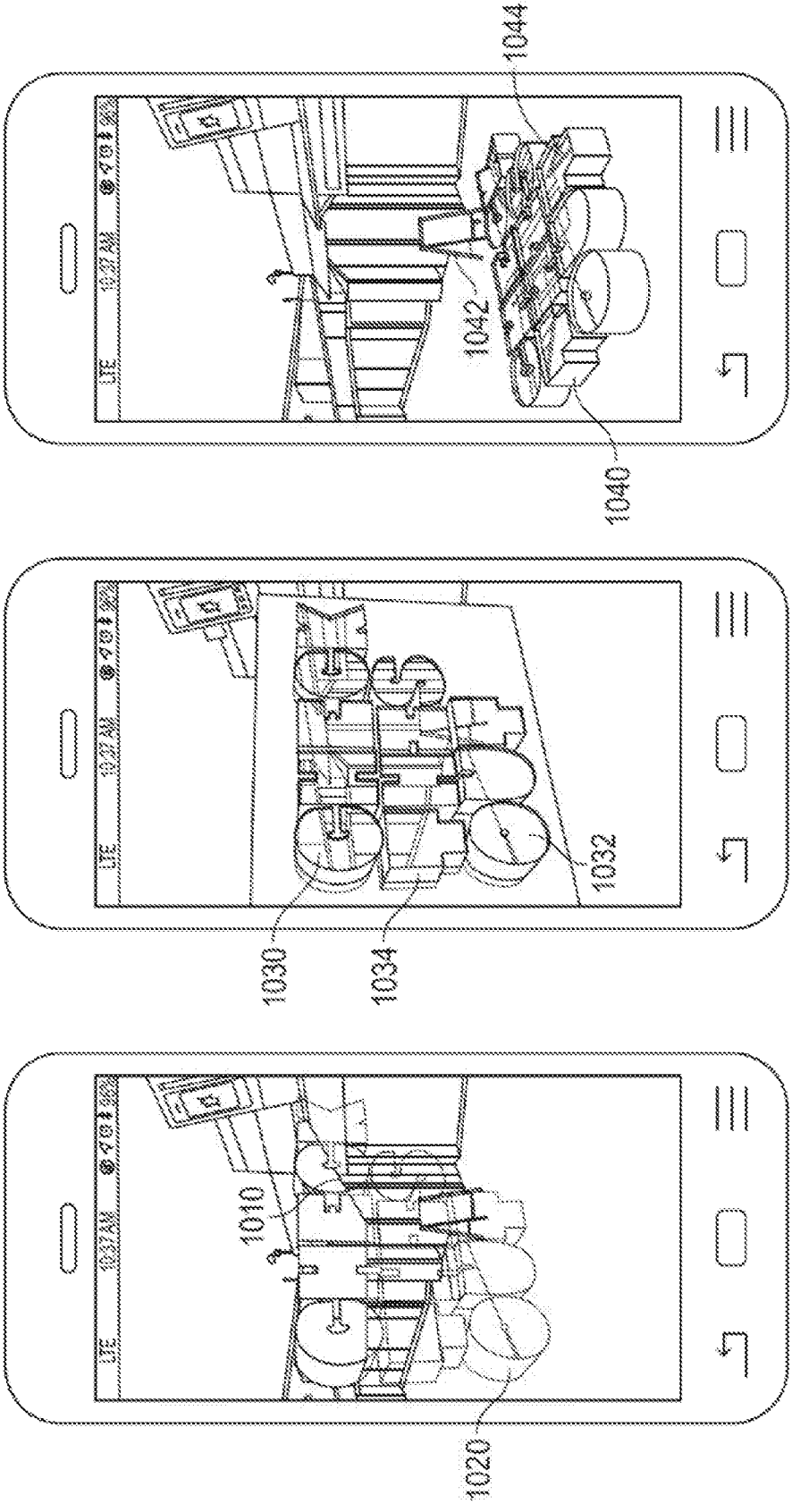


FIG. 10

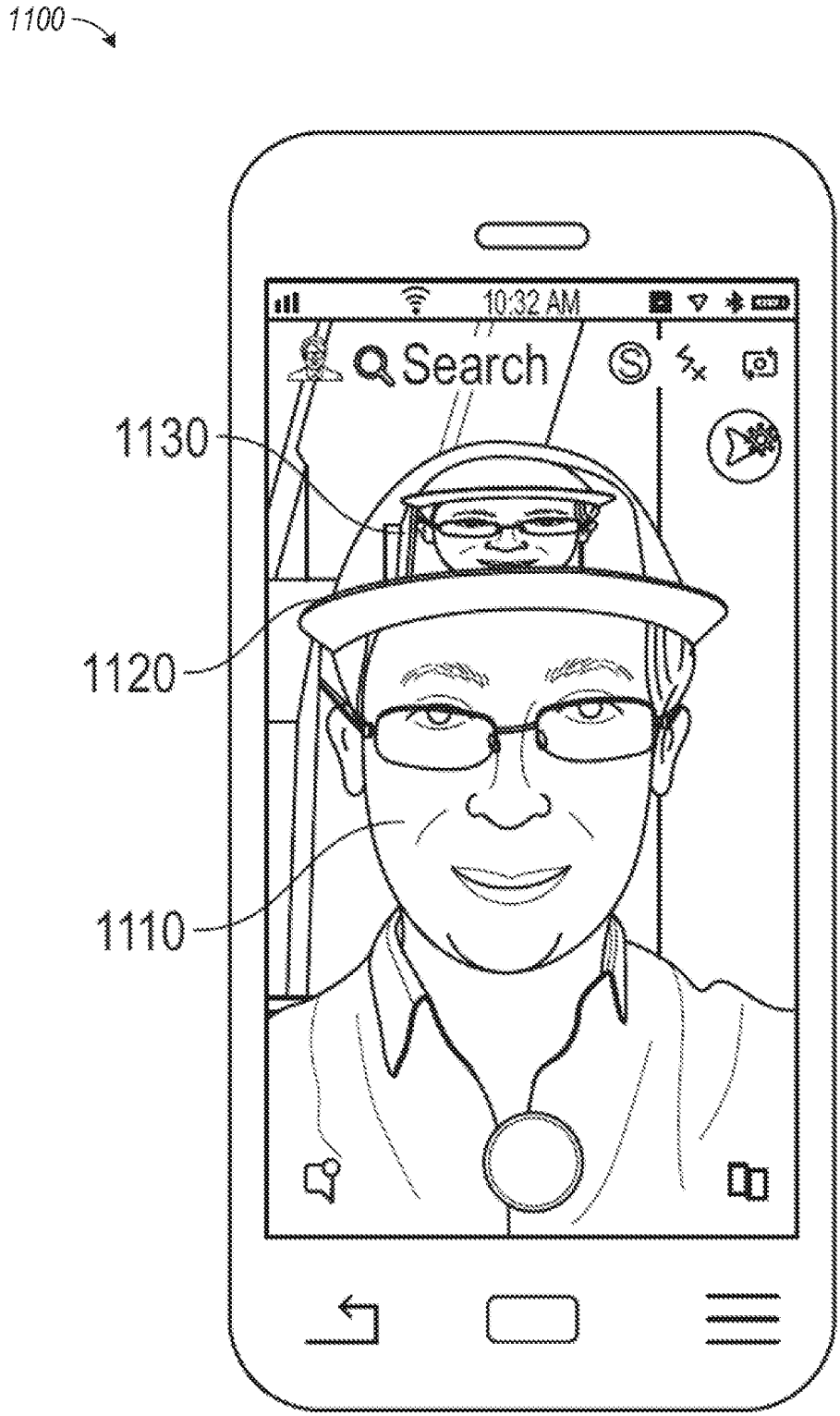


FIG. 11

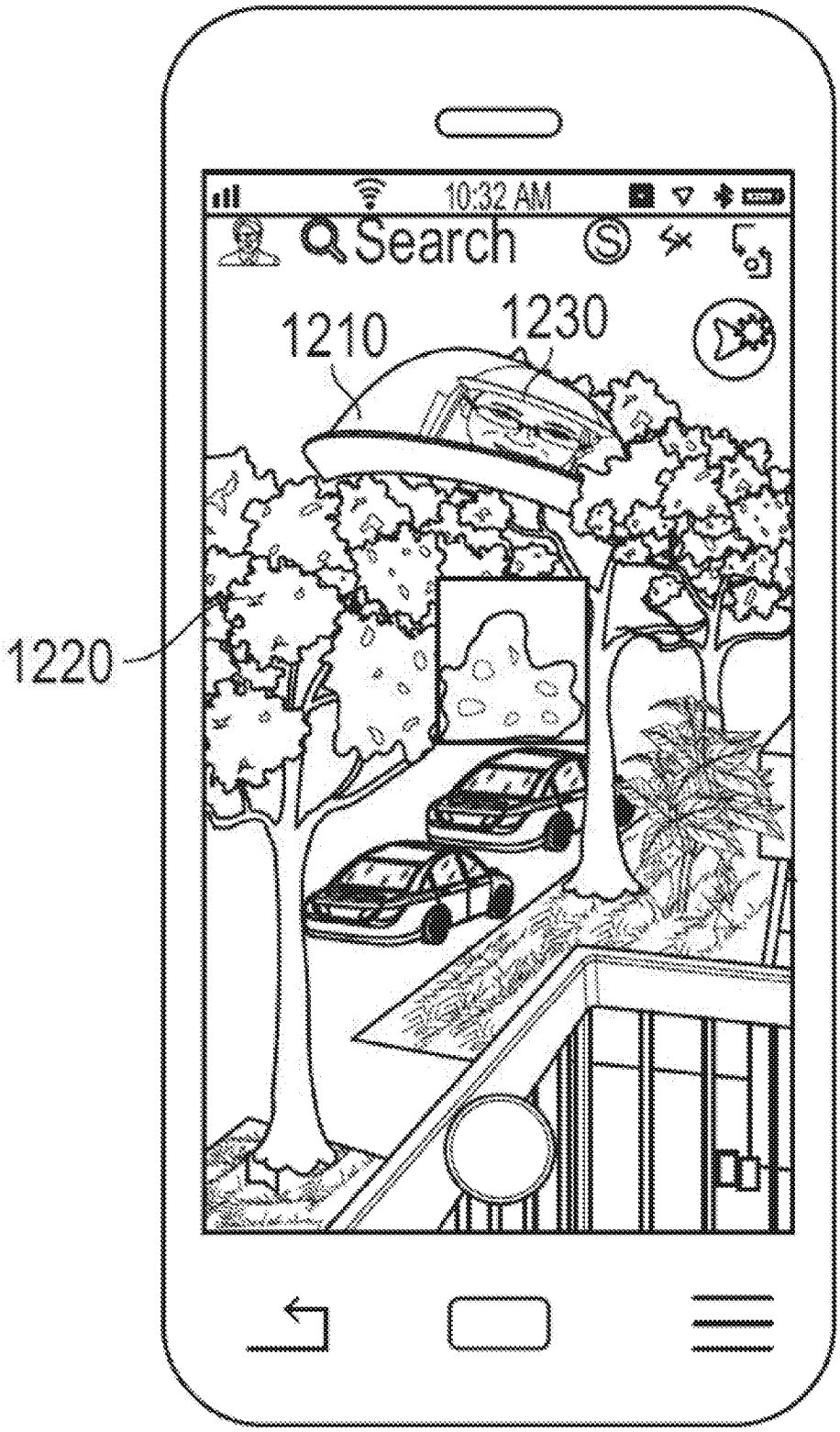


FIG. 12A

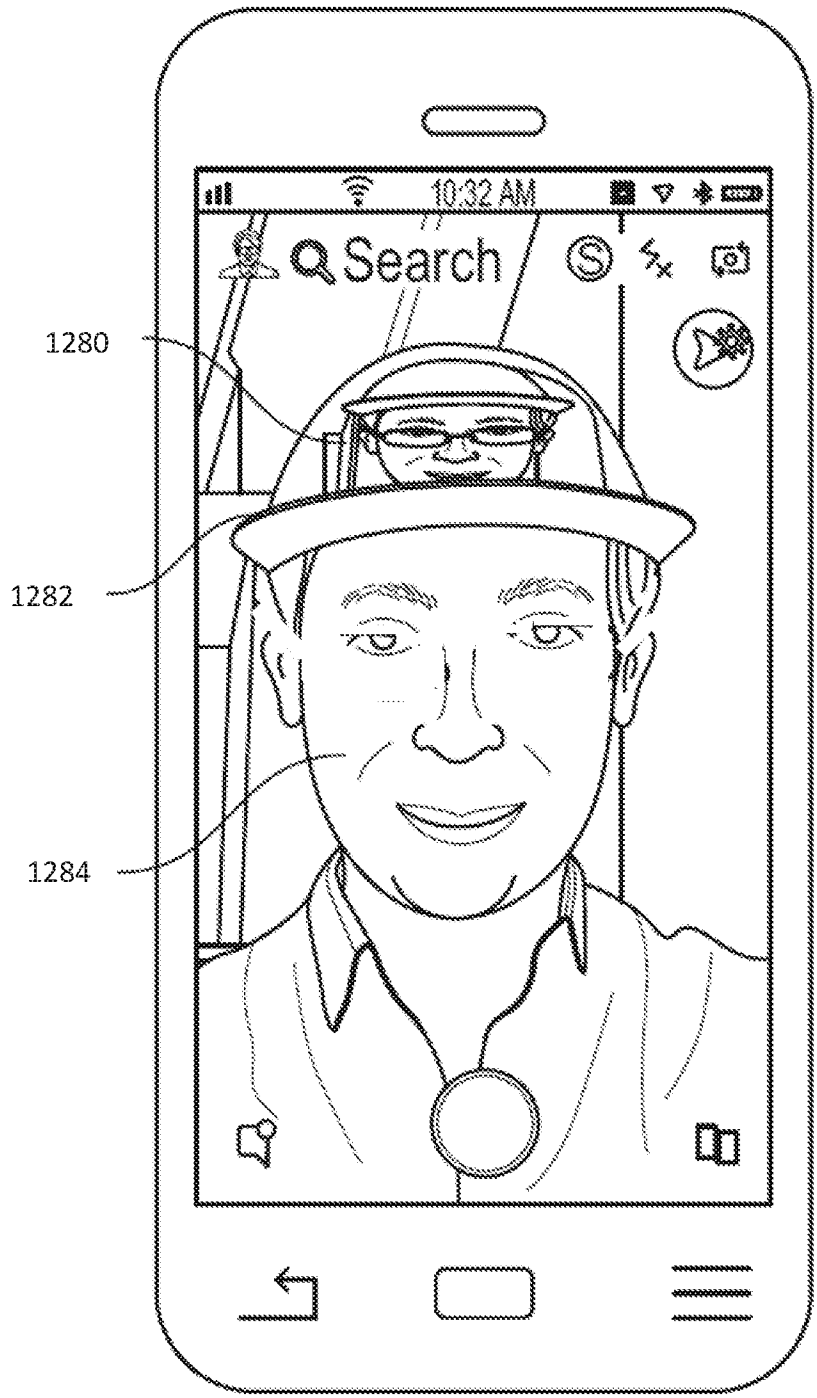


FIG. 12B

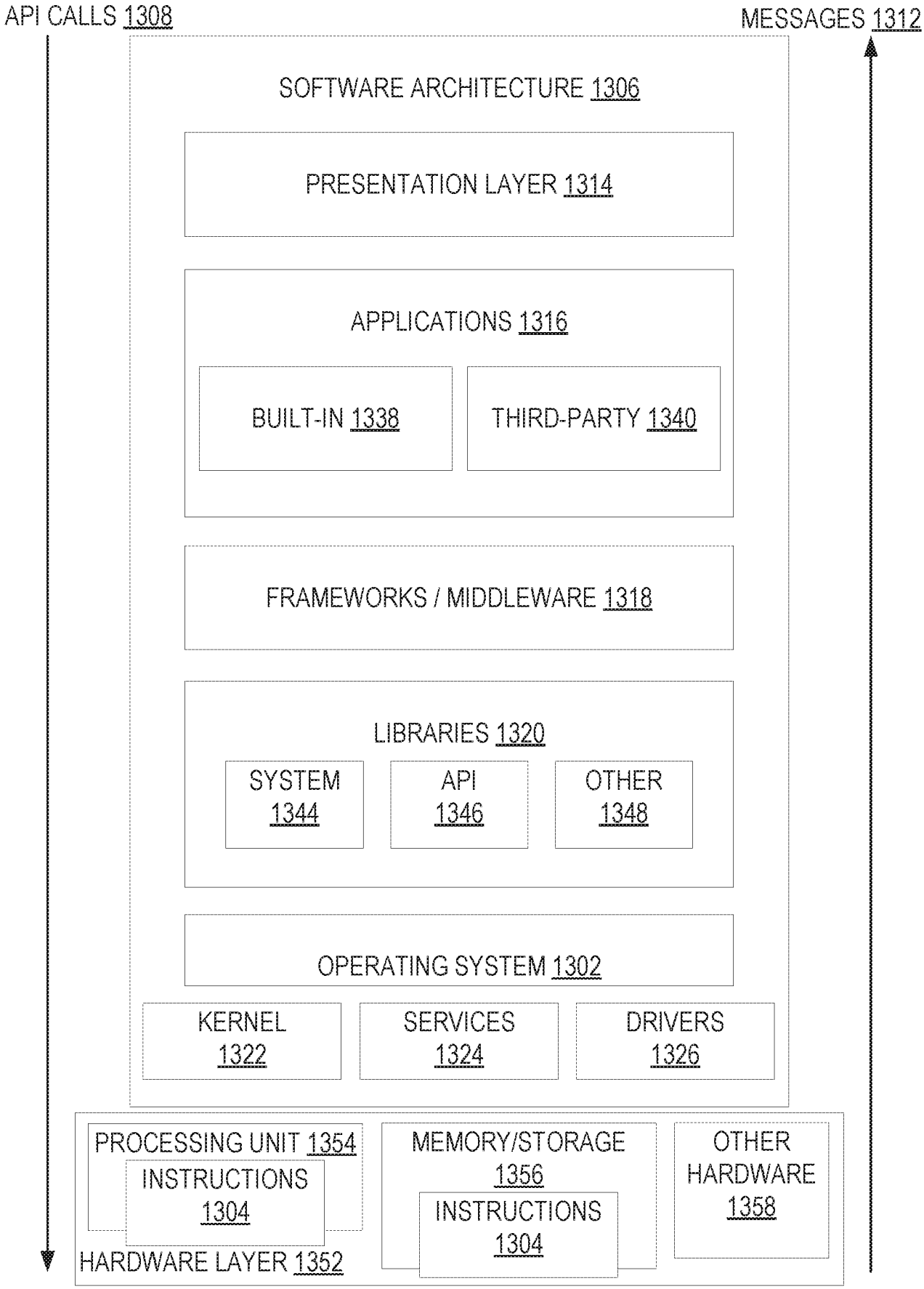

API CALLS 1308
MESSAGES 1312
SOFTWARE ARCHITECTURE 1306
PRESENTATION LAYER 1314
APPLICATIONS 1316
BUILT-IN 1338
THIRD-PARTY 1340
FRAMEWORKS / MIDDLEWARE 1318
LIBRARIES 1320
SYSTEM 1344
API 1346
OTHER 1348
OPERATING SYSTEM 1302
KERNEL 1322
SERVICES 1324
DRIVERS 1326
PROCESSING UNIT 1354
INSTRUCTIONS 1304
HARDWARE LAYER 1352
MEMORY/STORAGE 1356
INSTRUCTIONS 1304
OTHER HARDWARE 1358

FIG. 13

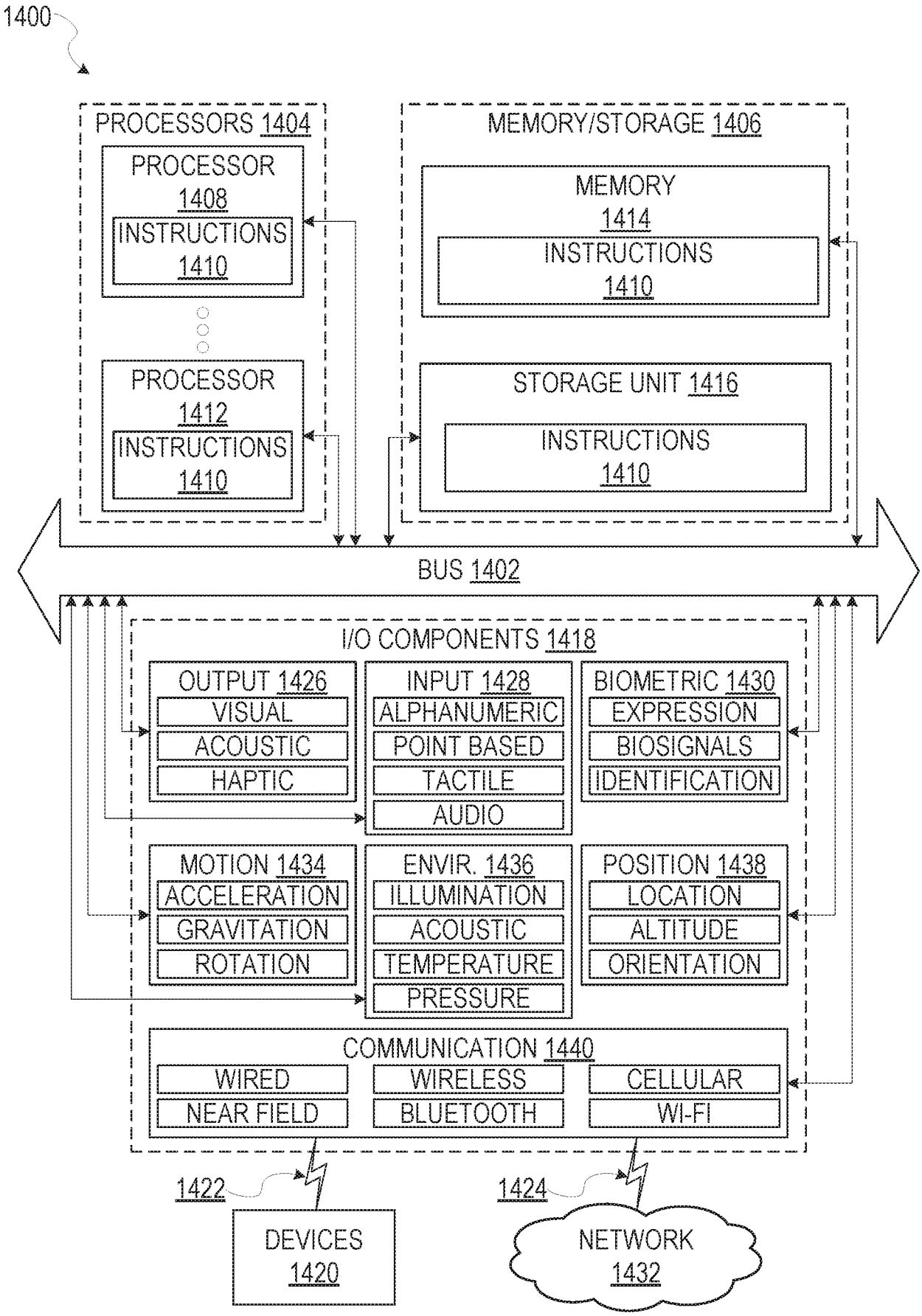

1400
PROCESSORS 1404
PROCESSOR 1408
INSTRUCTIONS 1410
PROCESSOR 1412
INSTRUCTIONS 1410
MEMORY/STORAGE 1406
MEMORY 1414
INSTRUCTIONS 1410
STORAGE UNIT 1416
INSTRUCTIONS 1410
BUS 1402
I/O COMPONENTS 1418
OUTPUT 1426
VISUAL
ACOUSTIC
HAPTIC
INPUT 1428
ALPHANUMERIC
POINT BASED
TACTILE
AUDIO
BIOMETRIC 1430
EXPRESSION
BIOSIGNALS
IDENTIFICATION
MOTION 1434
ACCELERATION
GRAVITATION
ROTATION
ENVIR. 1436
ILLUMINATION
ACOUSTIC
TEMPERATURE
PRESSURE
POSITION 1438
LOCATION
ALTITUDE
ORIENTATION
COMMUNICATION 1440
WIRED
NEAR FIELD
WIRELESS
BLUETOOTH
CELLULAR
WI-FI
1422
DEVICES 1420
1424
NETWORK 1432

FIG. 14