US012393324B2

(12) **United States Patent**
Velicodnii

(10) **Patent No.: US 12,393,324 B2**
(45) **Date of Patent: *Aug. 19, 2025**

(54) **INTERACTING WITH VISUAL CODES WITHIN MESSAGING SYSTEM**

(71) Applicant: **Snap Inc.**, Santa Monica, CA (US)

(72) Inventor: **Vadim Velicodnii**, London (GB)

(73) Assignee: **Snap Inc.**, Santa Monica, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **18/634,020**

(22) Filed: **Apr. 12, 2024**

(65) **Prior Publication Data**

US 2024/0281121 A1 Aug. 22, 2024

Related U.S. Application Data

(63) Continuation of application No. 17/820,674, filed on Aug. 18, 2022, now Pat. No. 11,983,390.

(51) **Int. Cl.**

G06F 3/0484	(2022.01)
G06F 3/0482	(2013.01)
H04L 51/07	(2022.01)

(52) **U.S. Cl.**

CPC ***G06F 3/0484*** (2013.01); ***G06F 3/0482*** (2013.01); ***H04L 51/07*** (2022.05)

(58) **Field of Classification Search**

CPC G06F 3/0484; G06F 3/0482; H04L 51/07

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

9,111,164	B1 *	8/2015	Anderton	G06V 10/245
10,380,394	B1	8/2019	Boyd et al.	
11,630,877	B1 *	4/2023	Cansizoglu	G06F 16/955 715/207

(Continued)

FOREIGN PATENT DOCUMENTS

WO	WO-2024039977	A1	2/2024

OTHER PUBLICATIONS

"U.S. Appl. No. 17/820,674, Non Final Office Action mailed Jul. 19, 2023".

(Continued)

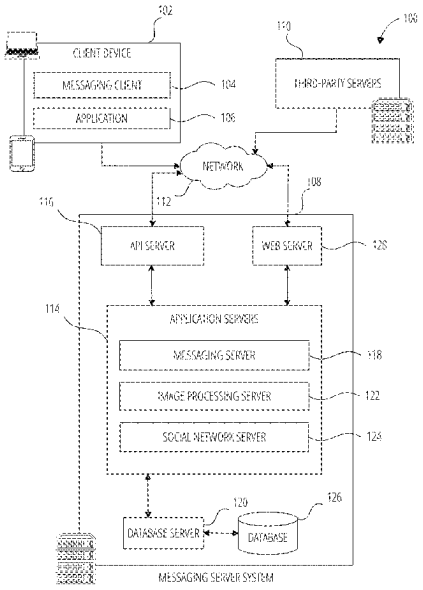
Primary Examiner — Cao H Nguyen

(74) *Attorney, Agent, or Firm* — Schwegman Lundberg & Woessner, P.A.

(57) **ABSTRACT**

Aspects of the present disclosure involve a system comprising a computer-readable storage medium storing a program and method for interacting with visual codes within a messaging system. The program and method provide for displaying, by a messaging application, captured image data comprising a visual code, the visual code including a custom graphic and being decodable to access a first feature of the messaging application; receiving user input selecting the visual code; displaying an updated version of the custom graphic; providing an animation which depicts the updated version of the custom graphic as moving from the visual code to an interface element comprising a group of icons, each icon within the group of icons being user-selectable to access a respective second feature of the messaging application; and updating the group of icons to include an additional icon which is user-selectable to access the first feature of the messaging application.

20 Claims, 12 Drawing Sheets



(56) **References Cited**

U.S. PATENT DOCUMENTS

2015/0070347	A1	3/2015	Hofmann et al.	
2019/0295056	A1*	9/2019	Wright	H04L 51/52
2021/0303077	A1	9/2021	Anvaripour et al.	
2021/0304505	A1	9/2021	Anvaripour et al.	
2021/0389850	A1*	12/2021	Charlton	G06F 8/36
2024/0061554	A1	2/2024	Velicodnil	

OTHER PUBLICATIONS

"U.S. Appl. No. 17/820,674, Notice of Allowance mailed Jan. 12, 2024", 11 pgs.

U.S. Appl. No. 17/820,674, Response filed Oct. 5, 2023 to Non Final Office Action mailed Jul. 19, 2023, 8 pgs.

"International Application Serial No. PCT/US2023/071671, International Search Report mailed Nov. 21, 2023", 4 pgs.

"International Application Serial No. PCT/US2023/071671, Written Opinion mailed Nov. 21, 2023", 7 pgs.

Zhan, Fangneng, et al., "EMLight: Lighting Estimation via Spherical Distribution Approximation", arXiv:2012.11116v1 [cs.CV], (Dec. 21, 2020), 9 pgs.

Zhan, Fangneng, et al., "GMLight: Lighting Estimation via Geometric Distribution Approximation", IEEE Transactions on Image Processing, arXiv:2102.10244v2 [cs.CV], (Jan. 5, 2022), 11 pgs.

* cited by examiner

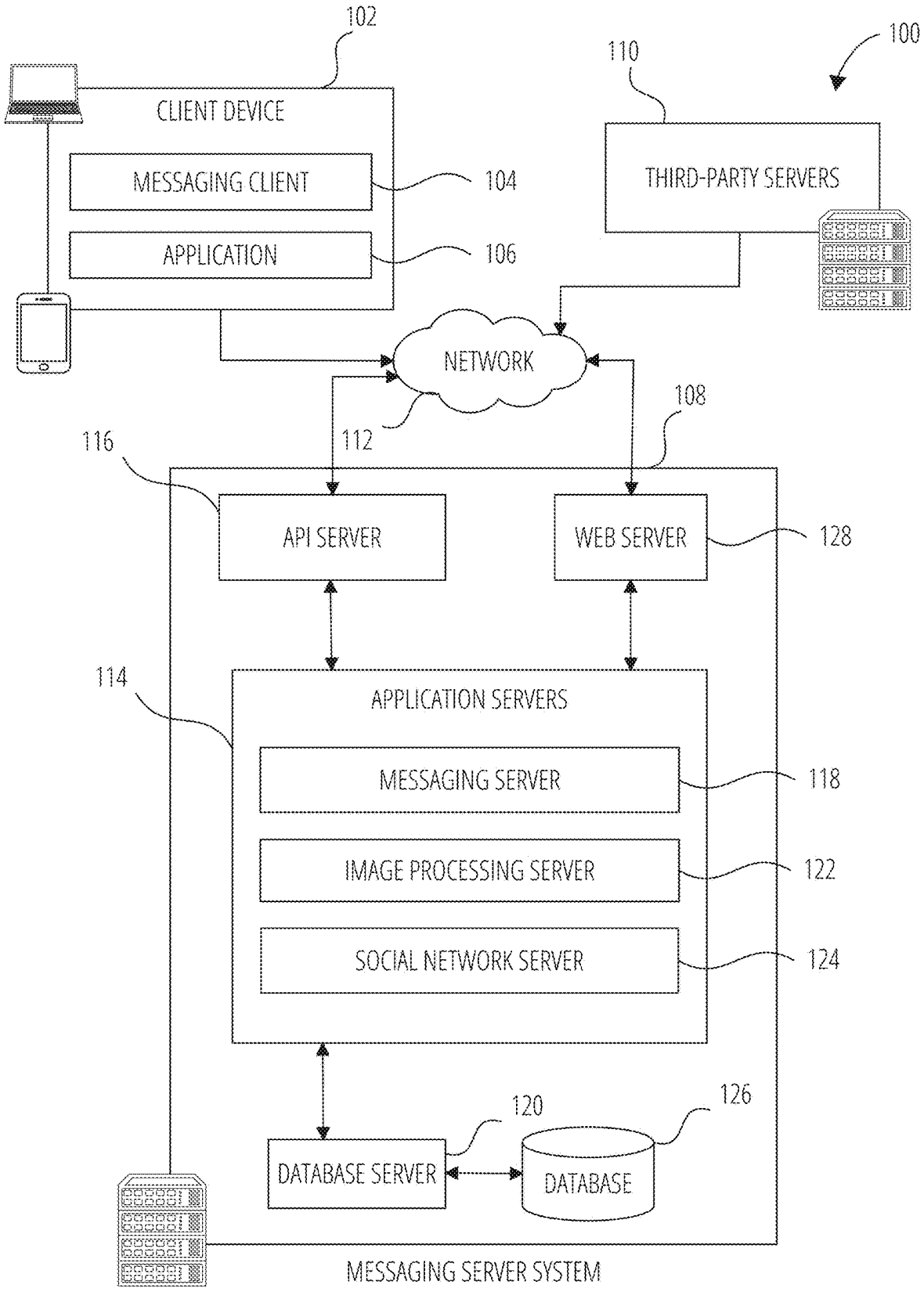

102
100
110
CLIENT DEVICE
MESSAGING CLIENT
104
APPLICATION
106
THIRD-PARTY SERVERS
NETWORK
108
112
116
API SERVER
WEB SERVER
128
114
APPLICATION SERVERS
MESSAGING SERVER
118
IMAGE PROCESSING SERVER
122
SOCIAL NETWORK SERVER
124
120
126
DATABASE SERVER
DATABASE
MESSAGING SERVER SYSTEM

FIG. 1

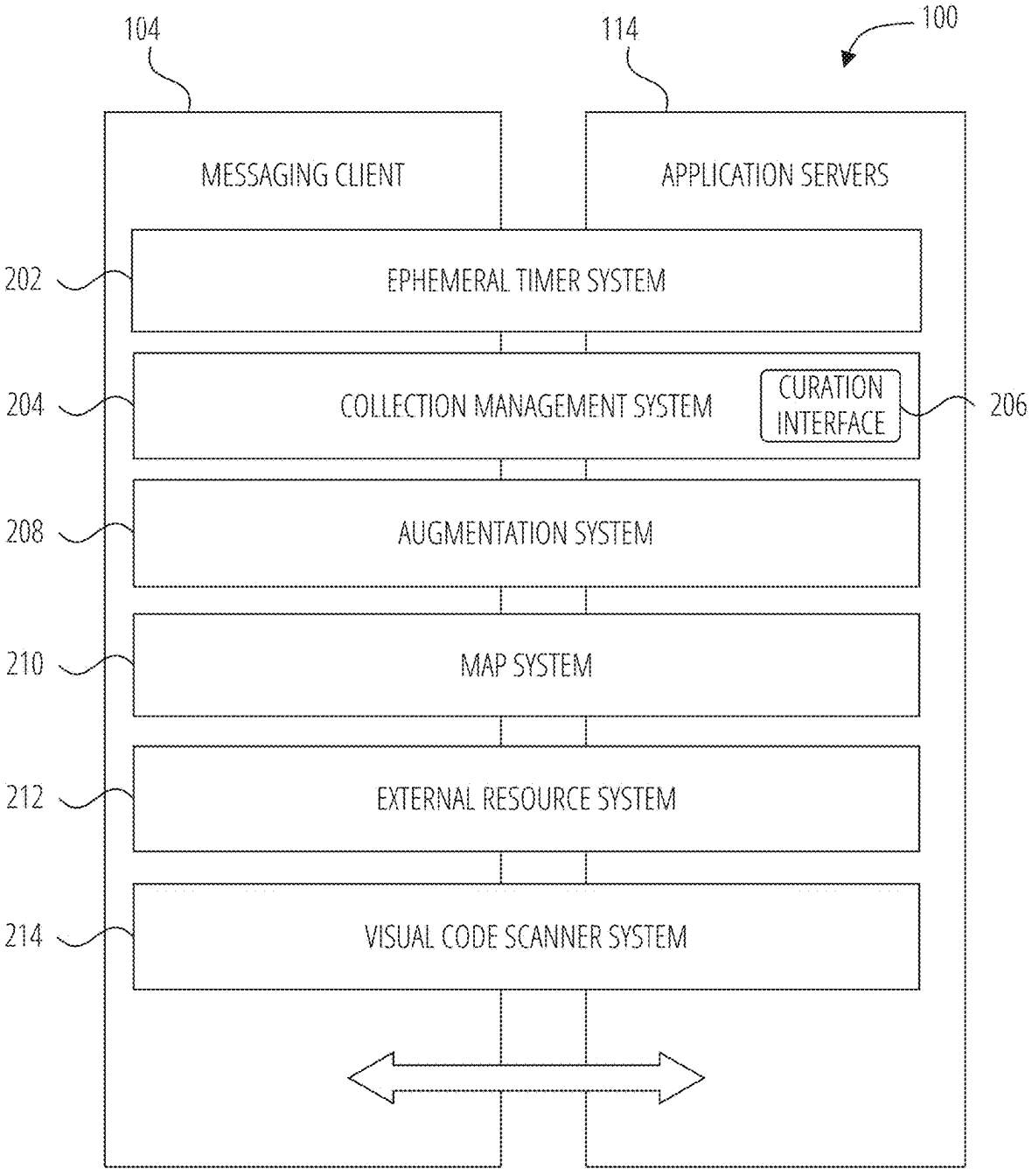

104
114
100
MESSAGING CLIENT
APPLICATION SERVERS
202
EPHEMERAL TIMER SYSTEM
204
COLLECTION MANAGEMENT SYSTEM
CURATION INTERFACE
206
208
AUGMENTATION SYSTEM
210
MAP SYSTEM
212
EXTERNAL RESOURCE SYSTEM
214
VISUAL CODE SCANNER SYSTEM

FIG. 2

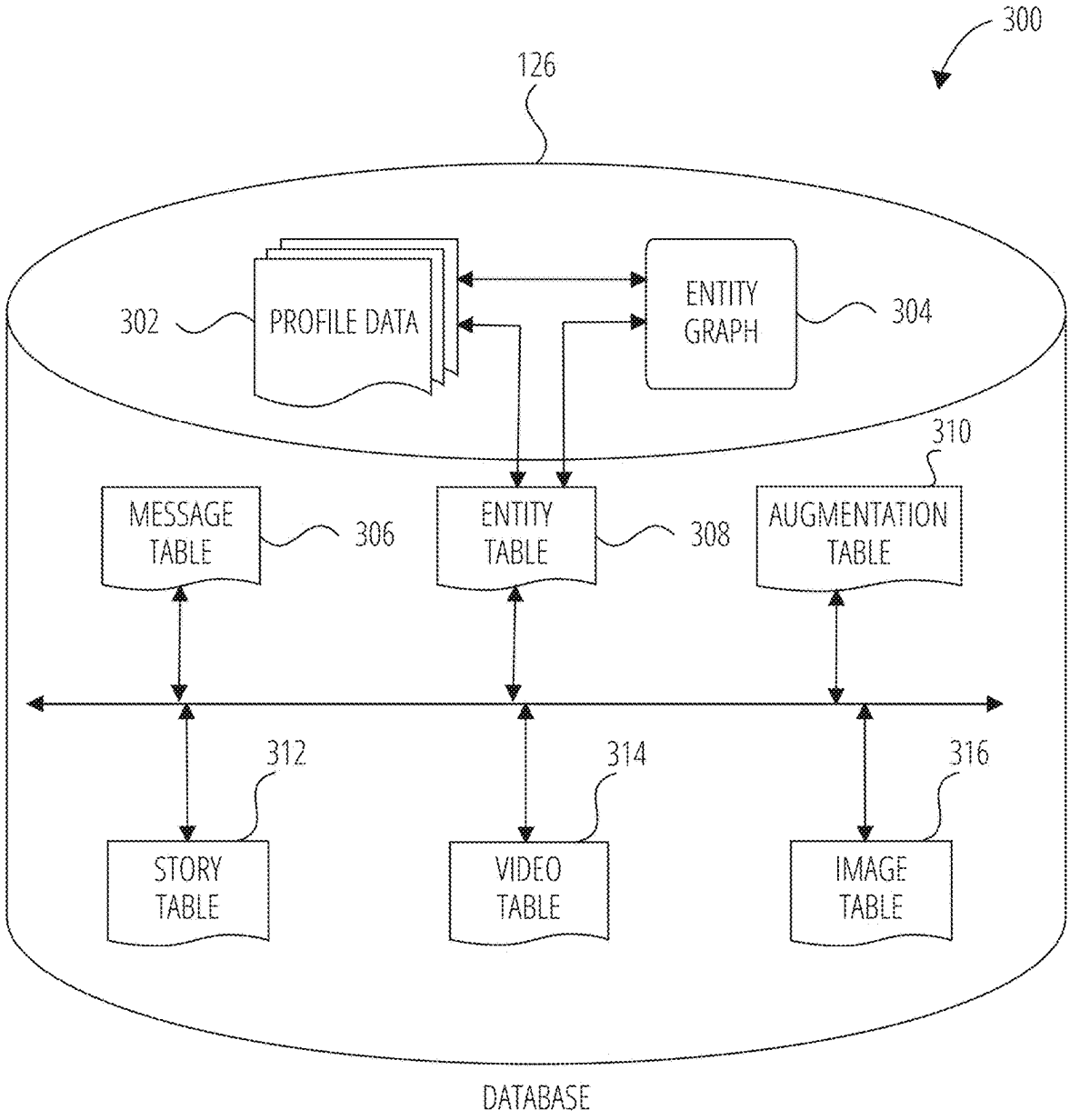

300
126
302
PROFILE DATA
ENTITY GRAPH
304
310
MESSAGE TABLE
306
ENTITY TABLE
308
AUGMENTATION TABLE
312
314
316
STORY TABLE
VIDEO TABLE
IMAGE TABLE
DATABASE

FIG. 3

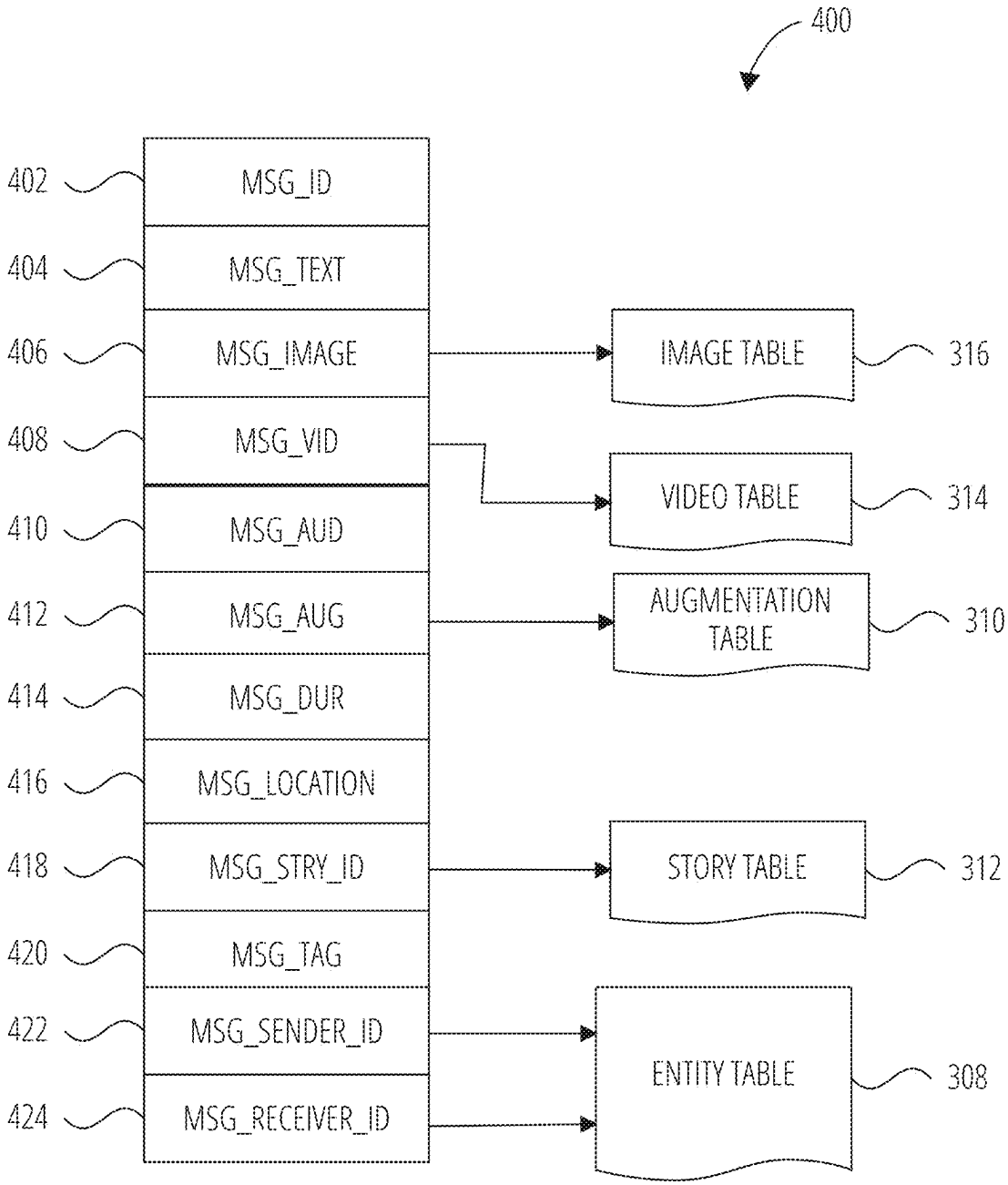

400
402
MSG_ID
404
MSG_TEXT
406
MSG_IMAGE
408
MSG_VID
410
MSG_AUD
412
MSG_AUG
414
MSG_DUR
416
MSG_LOCATION
418
MSG_STRY_ID
420
MSG_TAG
422
MSG_SENDER_ID
424
MSG_RECEIVER_ID
IMAGE TABLE
316
VIDEO TABLE
314
AUGMENTATION TABLE
310
STORY TABLE
312
ENTITY TABLE
308

FIG. 4

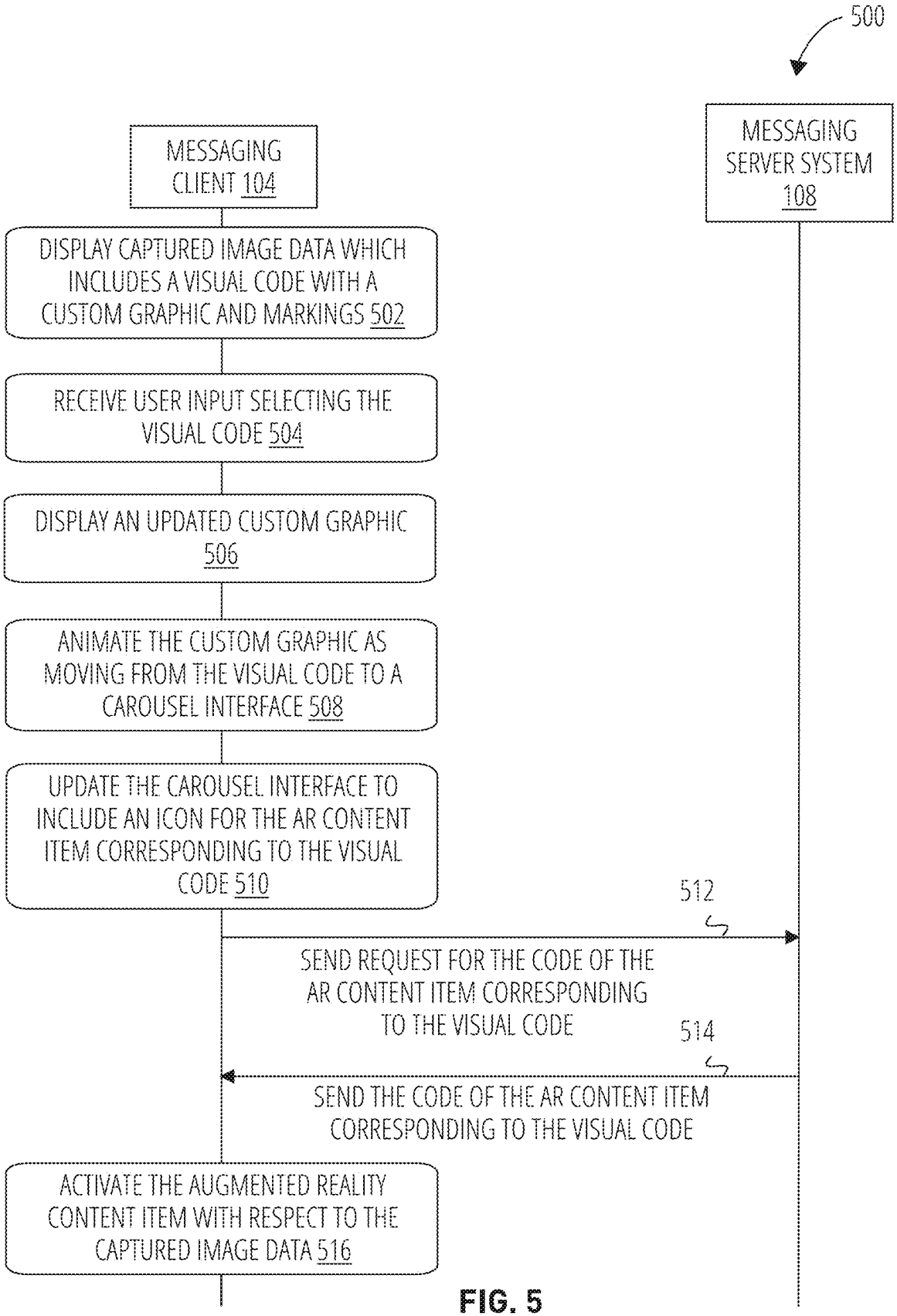

500
MESSAGING CLIENT 104
MESSAGING SERVER SYSTEM 108
DISPLAY CAPTURED IMAGE DATA WHICH INCLUDES A VISUAL CODE WITH A CUSTOM GRAPHIC AND MARKINGS 502
RECEIVE USER INPUT SELECTING THE VISUAL CODE 504
DISPLAY AN UPDATED CUSTOM GRAPHIC 506
ANIMATE THE CUSTOM GRAPHIC AS MOVING FROM THE VISUAL CODE TO A CAROUSEL INTERFACE 508
UPDATE THE CAROUSEL INTERFACE TO INCLUDE AN ICON FOR THE AR CONTENT ITEM CORRESPONDING TO THE VISUAL CODE 510
512
SEND REQUEST FOR THE CODE OF THE AR CONTENT ITEM CORRESPONDING TO THE VISUAL CODE
514
SEND THE CODE OF THE AR CONTENT ITEM CORRESPONDING TO THE VISUAL CODE
ACTIVATE THE AUGMENTED REALITY CONTENT ITEM WITH RESPECT TO THE CAPTURED IMAGE DATA 516

FIG. 5

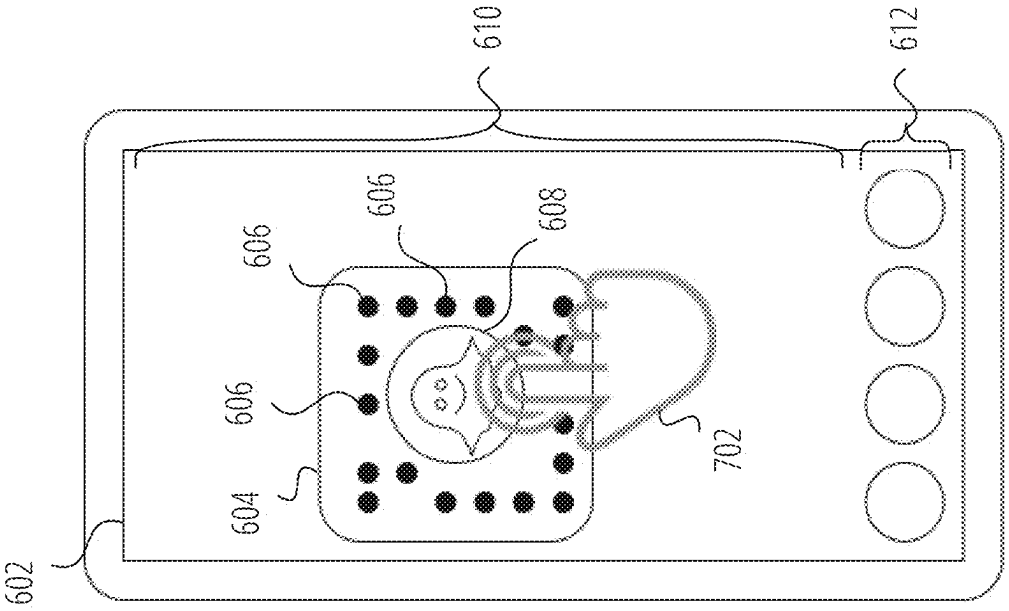

610
612
606
608
606
606
604
602
702

FIG. 7

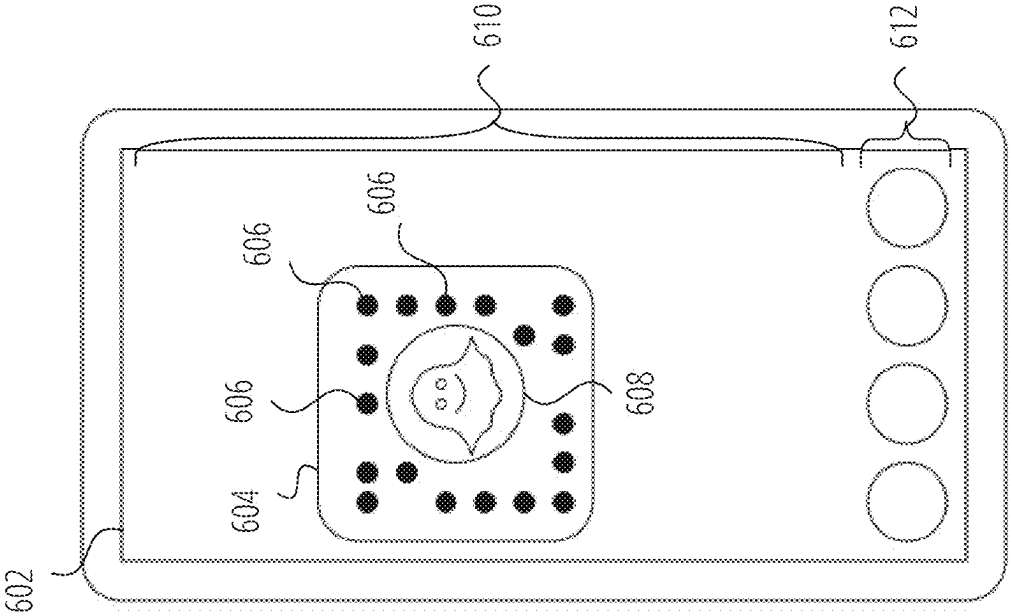

610
612
606
606
606
608
604
602

FIG. 6

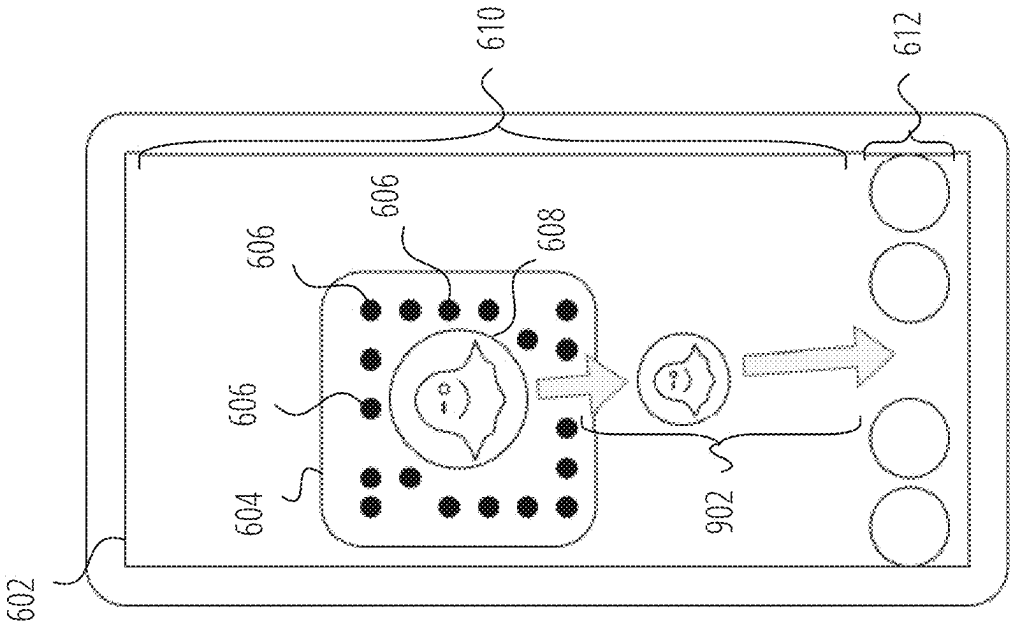

610
612
606
606
608
606
604
902
602

FIG. 9

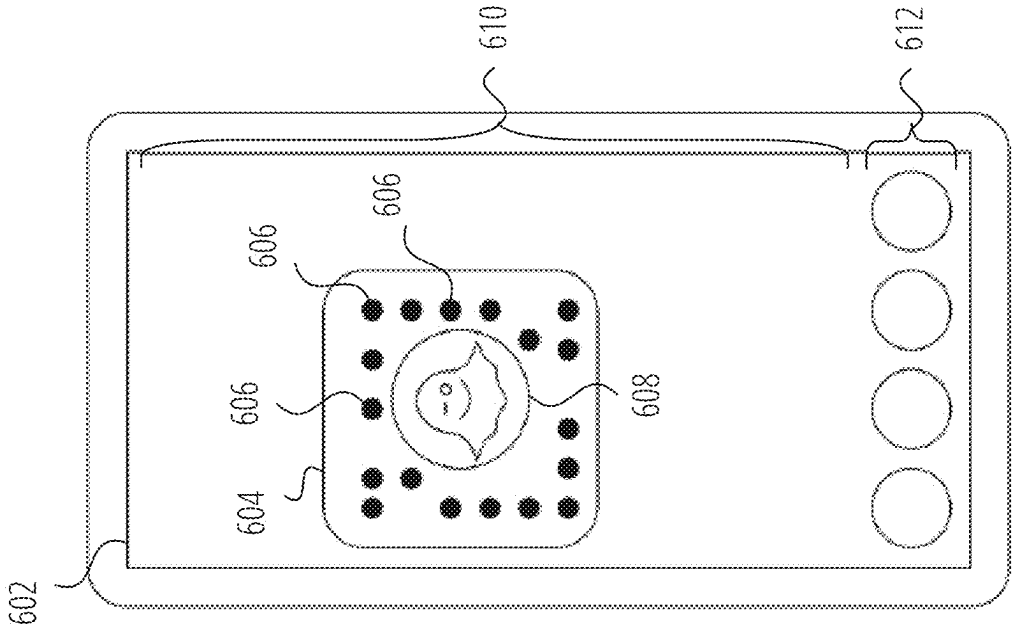

610
612
606
606
606
608
604
602

FIG. 8

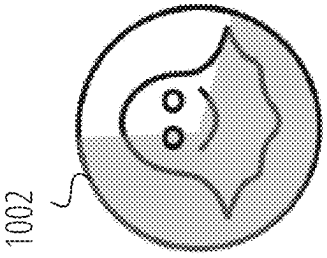


FIG. 10C

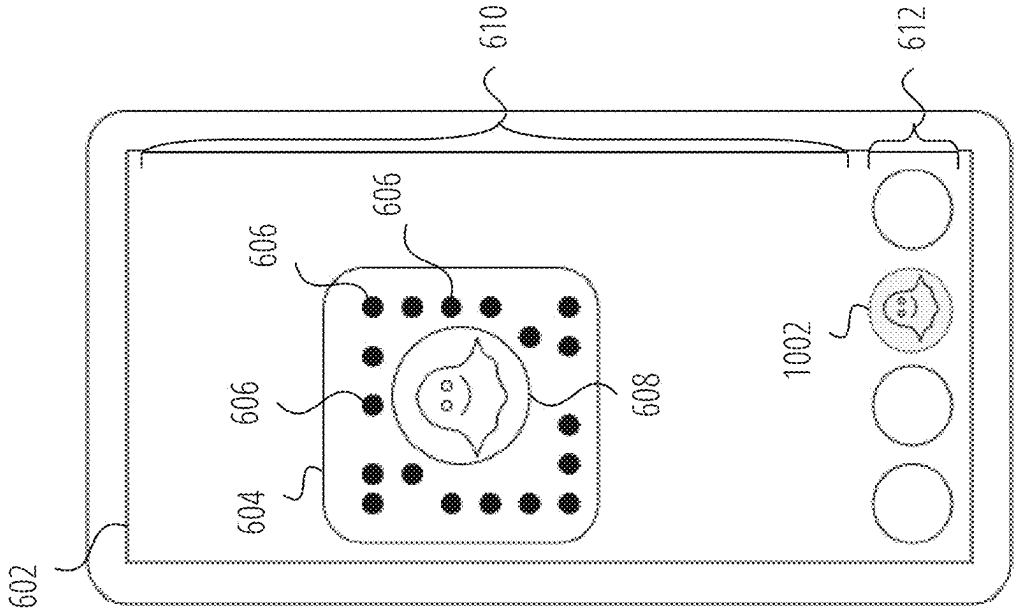


FIG. 10B

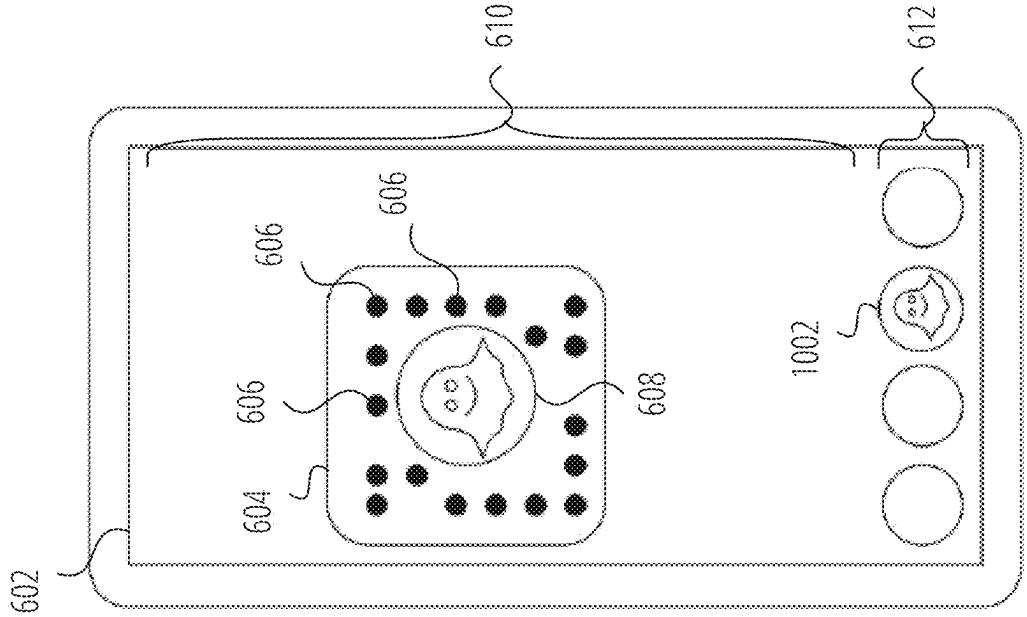


FIG. 10A

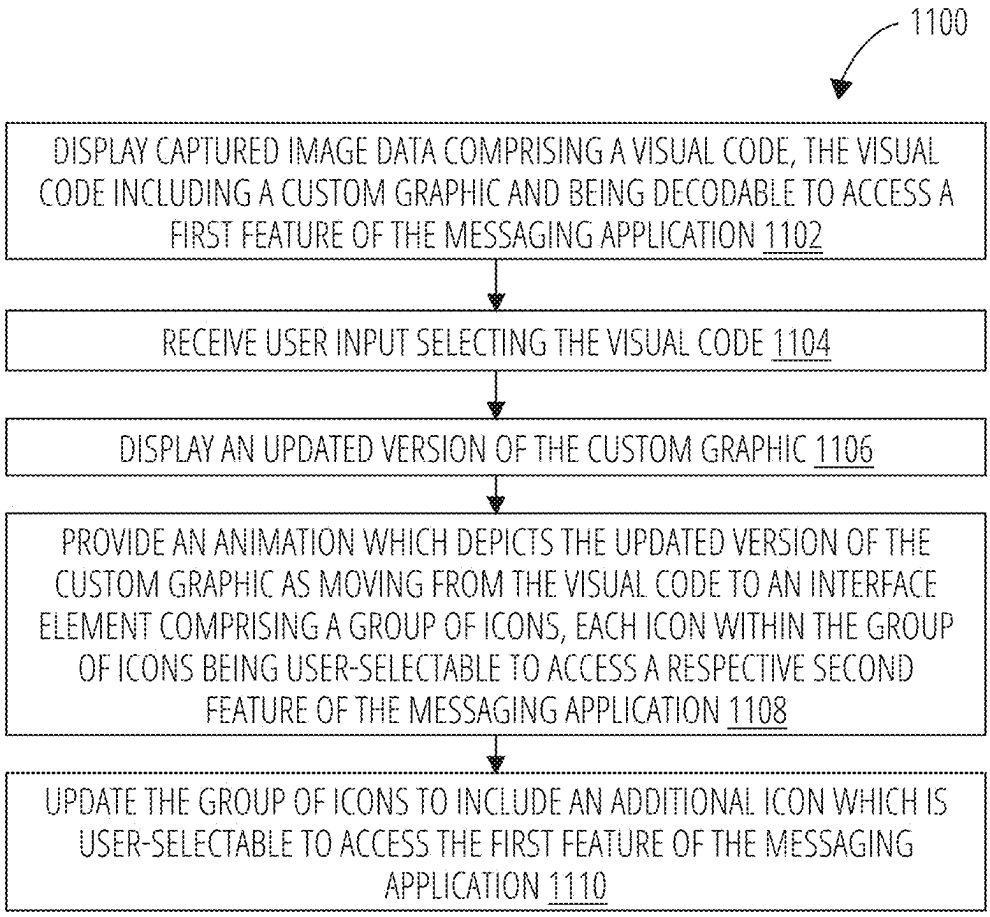

1100
DISPLAY CAPTURED IMAGE DATA COMPRISING A VISUAL CODE, THE VISUAL CODE INCLUDING A CUSTOM GRAPHIC AND BEING DECODABLE TO ACCESS A FIRST FEATURE OF THE MESSAGING APPLICATION 1102
RECEIVE USER INPUT SELECTING THE VISUAL CODE 1104
DISPLAY AN UPDATED VERSION OF THE CUSTOM GRAPHIC 1106
PROVIDE AN ANIMATION WHICH DEPICTS THE UPDATED VERSION OF THE CUSTOM GRAPHIC AS MOVING FROM THE VISUAL CODE TO AN INTERFACE ELEMENT COMPRISING A GROUP OF ICONS, EACH ICON WITHIN THE GROUP OF ICONS BEING USER-SELECTABLE TO ACCESS A RESPECTIVE SECOND FEATURE OF THE MESSAGING APPLICATION 1108
UPDATE THE GROUP OF ICONS TO INCLUDE AN ADDITIONAL ICON WHICH IS USER-SELECTABLE TO ACCESS THE FIRST FEATURE OF THE MESSAGING APPLICATION 1110

FIG. 11

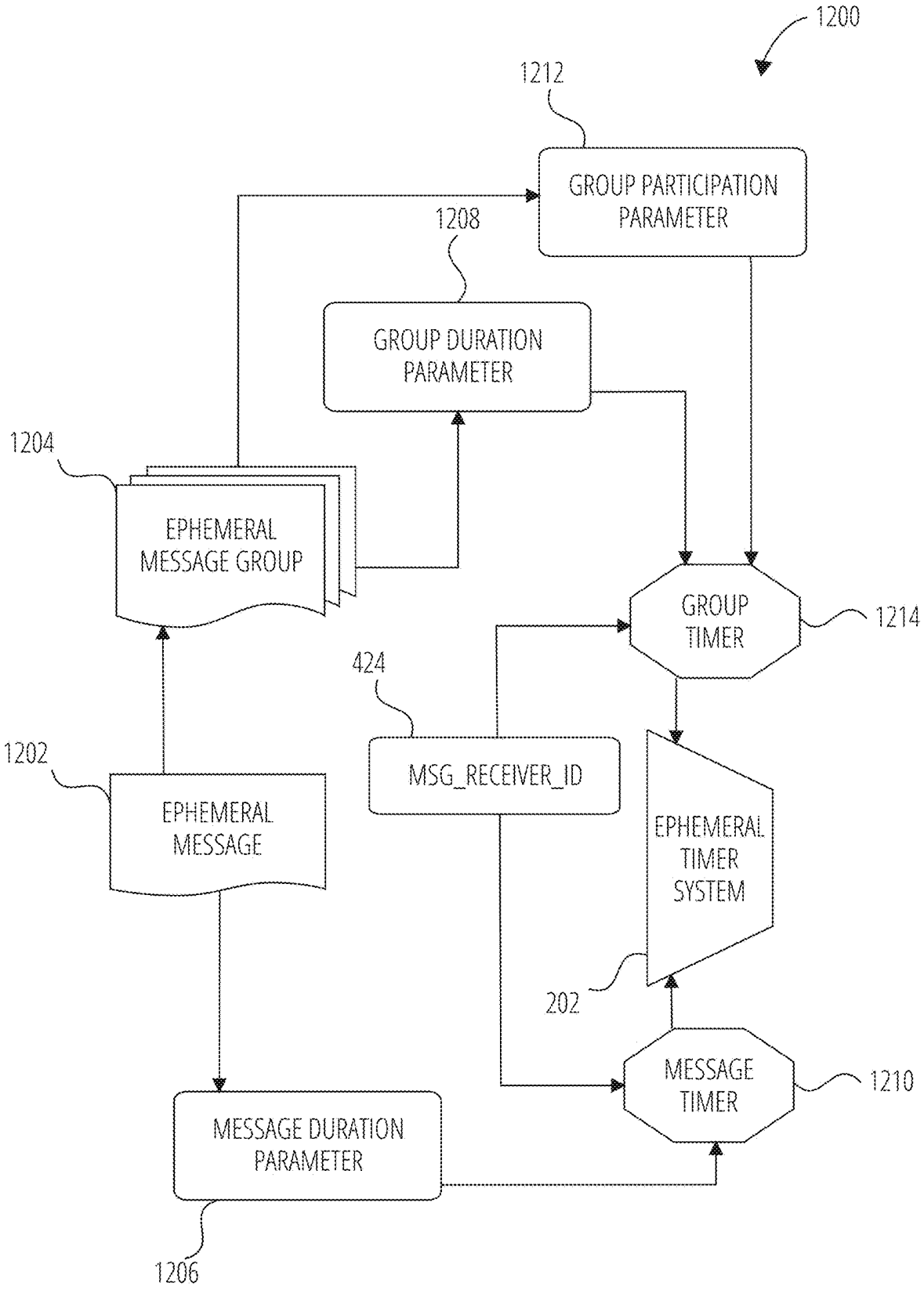

1200
1212
GROUP PARTICIPATION PARAMETER
1208
GROUP DURATION PARAMETER
1204
EPHEMERAL MESSAGE GROUP
GROUP TIMER
1214
424
MSG_RECEIVER_ID
1202
EPHEMERAL MESSAGE
EPHEMERAL TIMER SYSTEM
202
MESSAGE TIMER
1210
MESSAGE DURATION PARAMETER
1206

FIG. 12

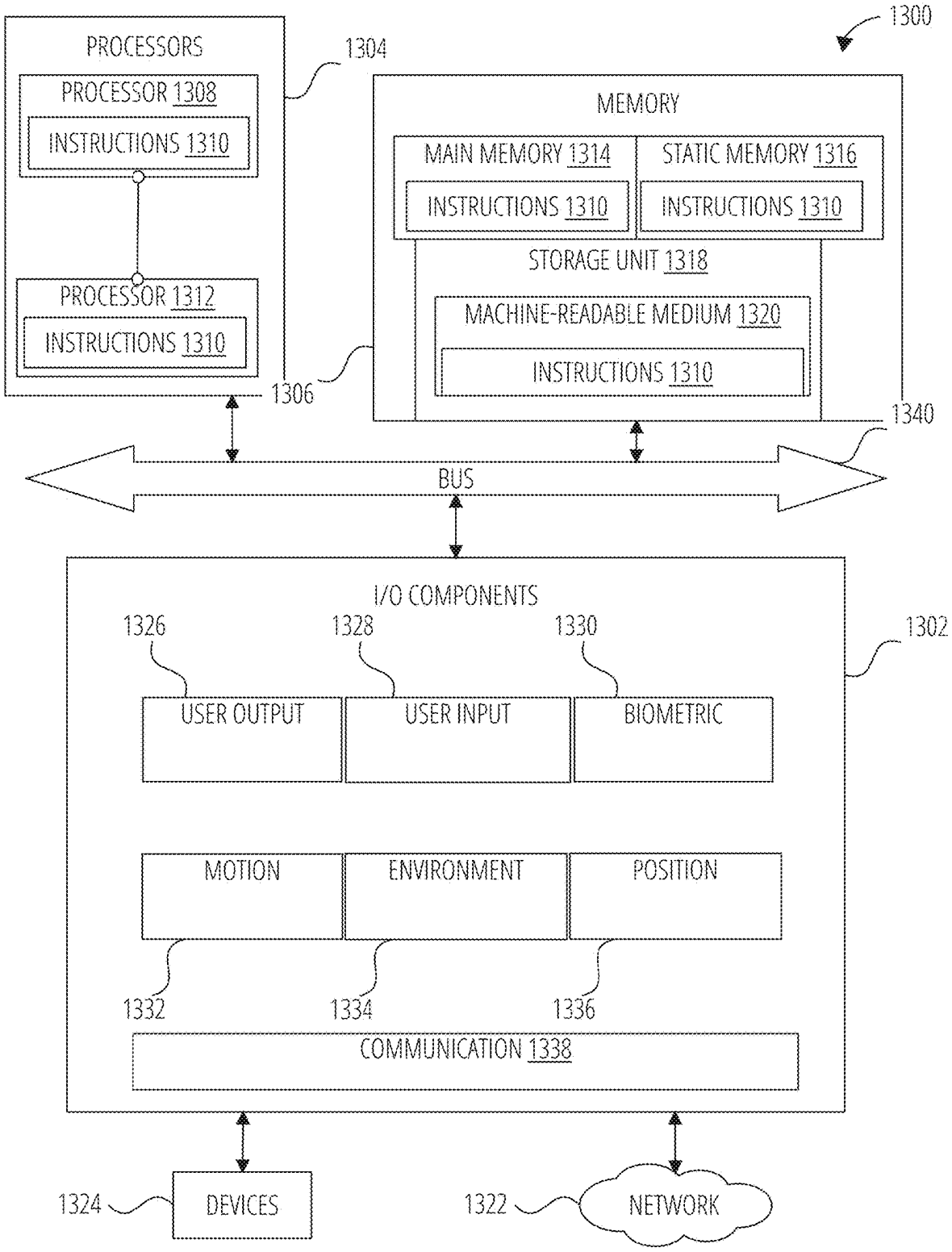

1300
PROCESSORS
1304
PROCESSOR 1308
INSTRUCTIONS 1310
PROCESSOR 1312
INSTRUCTIONS 1310
MEMORY
MAIN MEMORY 1314
INSTRUCTIONS 1310
STATIC MEMORY 1316
INSTRUCTIONS 1310
STORAGE UNIT 1318
MACHINE-READABLE MEDIUM 1320
INSTRUCTIONS 1310
1306
1340
BUS
I/O COMPONENTS
1326
1328
1330
1302
USER OUTPUT
USER INPUT
BIOMETRIC
MOTION
ENVIRONMENT
POSITION
1332
1334
1336
COMMUNICATION 1338
1324
DEVICES
1322
NETWORK

FIG. 13

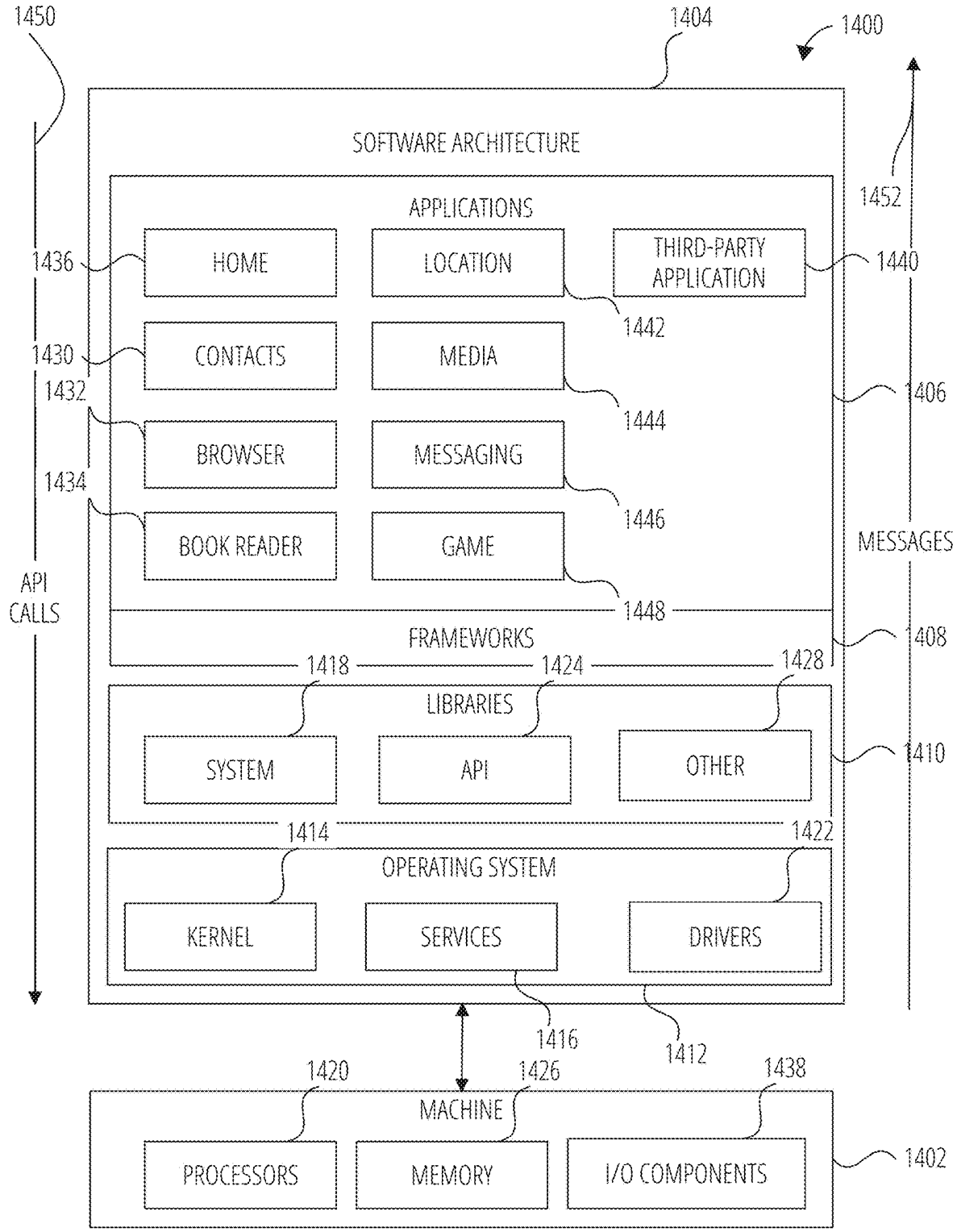

1450
1404
1400
SOFTWARE ARCHITECTURE
APPLICATIONS
1452
1436
HOME
LOCATION
THIRD-PARTY APPLICATION
1440
1442
1430
CONTACTS
MEDIA
1432
1444
1406
BROWSER
MESSAGING
1434
1446
BOOK READER
GAME
MESSAGES
API CALLS
1448
FRAMEWORKS
1408
1418
1424
1428
LIBRARIES
SYSTEM
API
OTHER
1410
1414
1422
OPERATING SYSTEM
KERNEL
SERVICES
DRIVERS
1416
1412
1420
1426
1438
MACHINE
PROCESSORS
MEMORY
I/O COMPONENTS
1402

FIG. 14

INTERACTING WITH VISUAL CODES WITHIN MESSAGING SYSTEM

CROSS-REFERENCE TO RELATED APPLICATION

This application is a continuation of U.S. patent application Ser. No. 17/820,674, filed Aug. 18, 2022, which is incorporated by reference herein in its entirety.

TECHNICAL FIELD

The present disclosure relates generally to a messaging system, including interacting with visual codes for accessing features within a messaging system.

BACKGROUND

Messaging systems provide for the exchange of message content between users. For example, a messaging system allows a user to exchange message content (e.g., text, images) with one or more other users.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

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 nonlimiting examples are illustrated in the figures of the accompanying drawings in which:

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

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

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

FIG. **4** is a diagrammatic representation of a message, in accordance with some examples.

FIG. **5** is an interaction diagram illustrating a process for interacting with visual codes for accessing features within a messaging system, in accordance with some examples.

FIG. **6** illustrates a user interface in which a visual code is displayed within captured image data, in accordance with some example embodiments.

FIG. **7** illustrates a user interface in a case where the user selects a visual code to activate an augmented reality content item, in accordance with some example embodiments.

FIG. **8** illustrates a user interface in which a custom graphic of a visual code is updated in response to user selection of the visual code, in accordance with some example embodiments.

FIG. **9** illustrates a user interface in which an updated custom graphic is animated to move toward and be inserted within a carousel interface, in accordance with some example embodiments.

FIGS. **10**A-**10**C illustrate user interfaces corresponding to different manners in which an augmented reality content item corresponding to a visual code is downloaded and activated, in accordance with some example embodiments.

FIG. **11** is a flowchart illustrating a process for interacting with visual codes for accessing features within a messaging system, in accordance with some examples.

FIG. **12** is a flowchart for an access-limiting process, in accordance with some examples.

FIG. **13** is a diagrammatic representation of a machine in the form of a computer system within which a set of instructions may be executed for causing the machine to perform any one or more of the methodologies discussed herein, in accordance with some examples.

FIG. **14** is a block diagram showing a software architecture within which examples may be implemented.

DETAILED DESCRIPTION

A messaging system typically allow users to exchange media content items (e.g., messages, images and/or video) with one another in a message thread. As described herein, a messaging system may provide visual codes QR code, 1Barcode, Data Matrix, Aztec code which are scannable for accessing particular features of the messaging system.

The disclosed embodiments relate to a messaging system for interacting with visual codes to access features of the messaging system. In other systems, after selecting the visual code, a user is presented with a separate message box (e.g., an overlay) that prompts the user to download and/or activate the feature. This message box may be distracting, as it switches the user's attention away from the current interface related to the feature.

The disclosed embodiments provide, in response to user selection of a visual code, a messaging client to update a custom graphic within the visual code. The messaging client animates movement of the custom graphic from the visual code area of the display to a carousel interface. The carousel interface includes user-selectable icons representing respective features of the messaging system. The messaging client updates the carousel interface to include an additional icon corresponding to the visual code, and downloads the code (e.g., assets) for accessing the corresponding feature. The download and activation of the feature may be automatic (e.g., without further prompts to the user), to reduce the amount of interaction required from end user.

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 instances of a client device **102**, each of which hosts a number of applications, including a messaging client **104** and other applications **106**. Each messaging client **104** is communicatively coupled to other instances of the messaging client **104** (e.g., hosted on respective other client devices **102**), a messaging server system **108** and third-party servers **110** via a network **112** (e.g., the Internet). A messaging client **104** can also communicate with locally-hosted applications **106** using Applications Program Interfaces (APIs).

A messaging client **104** is able to communicate and exchange data with other messaging clients **104** and with the messaging server system **108** via the network **112**. The data exchanged between messaging clients **104**, and between a messaging client **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). In some embodiments, the messaging client **104** activates a camera of the client device **102** (e.g., upon startup of the messaging client **104**).

The messaging server system **108** provides server-side functionality via the network **112** to a particular messaging

client **104**. While certain functions of the messaging system **100** are described herein as being performed by either a messaging client **104** or by the messaging server system **108**, the location of certain functionality either within the messaging client **104** or the messaging server system **108** may be 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 **104** where a client device **102** has sufficient processing capacity.

The messaging server system **108** supports various services and operations that are provided to the messaging client **104**. Such operations include transmitting data to, receiving data from, and processing data generated by the messaging client **104**. This data may include message content, client device information, geolocation information, media augmentation and overlays, message content persistence conditions, social network information, and live event information, as examples. Data exchanges within the messaging system **100** are invoked and controlled through functions available via user interfaces (UIs) of the messaging client **104**.

Turning now specifically to the messaging server system **108**, an Application Program Interface (API) server **116** is coupled to, and provides a programmatic interface to, application servers **114**. The application servers **114** are communicatively coupled to a database server **120**, which facilitates access to a database **126** that stores data associated with messages processed by the application servers **114**. Similarly, a web server **128** is coupled to the application servers **114**, and provides web-based interfaces to the application servers **114**. To this end, the web server **128** processes incoming network requests over the Hypertext Transfer Protocol (HTTP) and several other related protocols.

The Application Program Interface (API) server **116** receives and transmits message data (e.g., commands and message payloads) between the client device **102** and the application servers **114**. Specifically, the Application Program Interface (API) server **116** provides a set of interfaces (e.g., routines and protocols) that can be called or queried by the messaging client **104** in order to invoke functionality of the application servers **114**. The Application Program Interface (API) server **116** exposes various functions supported by the application servers **114**, including account registration, login functionality, the sending of messages, via the application servers **114**, from a particular messaging client **104** to another messaging client **104**, the sending of media files (e.g., images or video) from a messaging client **104** to a messaging server **118**, and for possible access by another messaging client **104**, the settings of a collection of media data (e.g., story), the retrieval of a list of friends of a user of a client device **102**, the retrieval of such collections, the retrieval of messages and content, the addition and deletion of entities (e.g., friends) to an entity graph (e.g., a social graph), the location of friends within a social graph, and opening an application event (e.g., relating to the messaging client **104**).

The application servers **114** host a number of server applications and subsystems, including for example a messaging server **118**, an image processing server **122**, and a social network server **124**. The messaging server **118** implements a number of message processing technologies and functions, particularly related to the aggregation and other processing of content (e.g., textual and multimedia content) included in messages received from multiple instances of the messaging client **104**. As will be described in further detail, the text and media content from multiple sources may be aggregated into collections of content (e.g., called stories or galleries). These collections are then made available to the messaging client **104**. Other processor and memory intensive processing of data may also be performed server-side by the messaging server **118**, in view of the hardware requirements for such processing.

The application servers **114** also include an image processing server **122** that is dedicated to performing various image processing operations, typically with respect to images or video within the payload of a message sent from or received at the messaging server **118**.

The social network server **124** supports various social networking functions and services and makes these functions and services available to the messaging server **118**. To this end, the social network server **124** maintains and accesses an entity graph **304** (as shown in FIG. **3**) within the database **126**. Examples of functions and services supported by the social network server **124** include the identification of other users of the 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.

Returning to the messaging client **104**, features and functions of an external resource (e.g., an application **106** or applet) are made available to a user via an interface of the messaging client **104**. In this context, "external" refers to the fact that the application **106** or applet is external to the messaging client **104**. The external resource is often provided by a third party but may also be provided by the creator or provider of the messaging client **104**. The messaging client **104** receives a user selection of an option to launch or access features of such an external resource. The external resource may be the application **106** installed on the client device **102** (e.g., a "native app"), or a small-scale version of the application (e.g., an "applet") that is hosted on the client device **102** or remote of the client device **102** (e.g., on third-party servers **110**). The small-scale version of the application includes a subset of features and functions of the application (e.g., the full-scale, native version of the application) and is implemented using a markup-language document. In one example, the small-scale version of the application (e.g., an "applet") is a web-based, markup-language version of the application and is embedded in the messaging client **104**. In addition to using markup-language documents (e.g., a .*ml file), an applet may incorporate a scripting language (e.g., a .*js file or a .json file) and a style sheet (e.g., a .*ss file).

In response to receiving a user selection of the option to launch or access features of the external resource, the messaging client **104** determines whether the selected external resource is a web-based external resource or a locally-installed application **106**. In some cases, applications **106** that are locally installed on the client device **102** can be launched independently of and separately from the messaging client **104**, such as by selecting an icon, corresponding to the application **106**, on a home screen of the client device **102**. Small-scale versions of such applications can be launched or accessed via the messaging client **104** and, in some examples, no or limited portions of the small-scale application can be accessed outside of the messaging client **104**. The small-scale application can be launched by the messaging client **104** receiving, from a third-party server **110** for example, a markup-language document associated with the small-scale application and processing such a document.

In response to determining that the external resource is a locally-installed application **106**, the messaging client **104** instructs the client device **102** to launch the external resource by executing locally-stored code corresponding to the external resource. In response to determining that the external resource is a web-based resource, the messaging client **104** communicates with the third-party servers **110** (for example) to obtain a markup-language document corresponding to the selected external resource. The messaging client **104** then processes the obtained markup-language document to present the web-based external resource within a user interface of the messaging client **104**.

The messaging client **104** can notify a user of the client device **102**, or other users related to such a user (e.g., "friends"), of activity taking place in one or more external resources. For example, the messaging client **104** can provide participants in a conversation (e.g., a chat session) in the messaging client **104** with notifications relating to the current or recent use of an external resource by one or more members of a group of users. One or more users can be invited to join in an active external resource or to launch a recently-used but currently inactive (in the group of friends) external resource. The external resource can provide participants in a conversation, each using respective messaging clients **104**, with the ability to share an item, status, state, or location in an external resource with one or more members of a group of users into a chat session. The shared item may be an interactive chat card with which members of the chat can interact, for example, to launch the corresponding external resource, view specific information within the external resource, or take the member of the chat to a specific location or state within the external resource. Within a given external resource, response messages can be sent to users on the messaging client **104**. The external resource can selectively include different media items in the responses, based on a current context of the external resource.

The messaging client **104** can present a list of the available external resources (e.g., applications **106** or applets) to a user to launch or access a given external resource. This list can be presented in a context-sensitive menu. For example, the icons representing different ones of the application **106** (or applets) can vary based on how the menu is launched by the user (e.g., from a conversation interface or from a non-conversation interface).

FIG. **2** is a block diagram illustrating further details regarding the messaging system **100**, according to some examples. Specifically, the messaging system **100** is shown to comprise the messaging client **104** and the application servers **114**. The messaging system **100** embodies a number of subsystems, which are supported on the client-side by the messaging client **104** and on the server-side by the application servers **114**. These subsystems include, for example, an ephemeral timer system **202**, a collection management system **204**, an augmentation system **208**, a map system **210**, an external resource system **212**, and a visual code scanner system **214**.

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

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

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

The augmentation system **208** provides various functions that enable a user to augment (e.g., annotate or otherwise modify or edit) media content associated with a message. For example, the augmentation system **208** provides functions related to the generation and publishing of media overlays for messages processed by the messaging system **100**. The augmentation system **208** operatively supplies a media overlay or augmentation (e.g., an image filter) to the messaging client **104** based on a geolocation of the client device **102**. In another example, the augmentation system **208** operatively supplies a media overlay to the messaging client **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. 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 may include text or image that can be overlaid on top of a photograph taken 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). In another example, the augmentation system **208** uses the geolocation of the client device **102** to identify a media overlay that includes the name of a merchant at the geolocation of the client device **102**. The media overlay may include other indicia associated with the merchant. The media overlays may be stored in the database **126** and accessed through the database server **120**.

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

In other examples, the augmentation system **208** provides a merchant-based publication platform that enables merchants to select a particular media overlay associated with a geolocation via a bidding process. For example, the augmentation system **208** associates the media overlay of the highest bidding merchant with a corresponding geolocation for a predefined amount of time.

In other examples, as discussed below with respect to FIG. **3**, the augmentation system **208** provides for presenting augmented reality content in association with an image or a video captured by a camera of the client device **102**. The augmentation system **208** may implement or otherwise access augmented reality content items (e.g., corresponding to applying Lenses or augmented reality experiences) for providing real-time special effect(s) and/or sound(s) that may be added to the image or video.

The map system **210** provides various geographic location functions, and supports the presentation of map-based media content and messages by the messaging client **104**. For example, the map system **210** enables the display of user icons or avatars (e.g., stored in profile data **302**) on a map to indicate a current or past location of "friends" of a user, as well as media content (e.g., collections of messages including photographs and videos) generated by such friends, within the context of a map. For example, a message posted by a user to the messaging system **100** from a specific geographic location may be displayed within the context of a map at that particular location to "friends" of a specific user on a map interface of the messaging client **104**. A user can furthermore share his or her location and status information (e.g., using an appropriate status avatar) with other users of the messaging system **100** via the messaging client **104**, with this location and status information being similarly displayed within the context of a map interface of the messaging client **104** to selected users.

The external resource system **212** provides an interface for the messaging client **104** to communicate with remote servers (e.g. third-party servers **110**) to launch or access external resources, i.e. applications or applets. Each third-party server **110** hosts, for example, a markup language (e.g., HTML5) based application or small-scale version of an application (e.g., game, utility, payment, or ride-sharing application). The messaging client **104** may launches a web-based resource (e.g., application) by accessing the HTML5 file from the third-party servers **110** associated with the web-based resource. In certain examples, applications hosted by third-party servers **110** are programmed in JavaScript leveraging a Software Development Kit (SDK) provided by the messaging server **118**. The SDK includes Application Programming Interfaces (APIs) with functions that can be called or invoked by the web-based application. In certain examples, the messaging server **118** includes a JavaScript library that provides a given external resource access to certain user data of the messaging client **104**. HTML5 is used as an example technology for programming games, but applications and resources programmed based on other technologies can be used.

In order to integrate the functions of the SDK into the web-based resource, the SDK is downloaded by a third-party server **110** from the messaging server **118** or is otherwise received by the third-party server **110**. Once downloaded or received, the SDK is included as part of the application code of a web-based external resource. The code of the web-based resource can then call or invoke certain functions of the SDK to integrate features of the messaging client **104** into the web-based resource.

The SDK stored on the messaging server **118** effectively provides the bridge between an external resource (e.g., applications **106** or applets and the messaging client **104**. This provides the user with a seamless experience of communicating with other users on the messaging client **104**, while also preserving the look and feel of the messaging client **104**. To bridge communications between an external resource and a messaging client **104**, in certain examples, the SDK facilitates communication between third-party servers **110** and the messaging client **104**. In certain examples, a WebViewJavaScriptBridge running on a client device **102** establishes two one-way communication channels between an external resource and the messaging client **104**. Messages are sent between the external resource and the messaging client **104** via these communication channels asynchronously. Each SDK function invocation is sent as a message and callback. Each SDK function is implemented by constructing a unique callback identifier and sending a message with that callback identifier.

By using the SDK, not all information from the messaging client **104** is shared with third-party servers **110**. The SDK limits which information is shared based on the needs of the external resource. In certain examples, each third-party server **110** provides an HTML5 file corresponding to the web-based external resource to the messaging server **118**. The messaging server **118** can add a visual representation (such as a box art or other graphic) of the web-based external resource in the messaging client **104**. Once the user selects the visual representation or instructs the messaging client **104** through a GUI of the messaging client **104** to access features of the web-based external resource, the messaging client **104** obtains the HTML5 file and instantiates the resources necessary to access the features of the web-based external resource.

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

The messaging client **104** controls the type of user data that is shared with external resources based on the type of external resource being authorized. For example, external resources that include full-scale applications (e.g., an application **106**) are provided with access to a first type of user

data (e.g., only two-dimensional avatars of users with or without different avatar characteristics). As another example, external resources that include small-scale versions of applications (e.g., web-based versions of applications) are provided with access to a second type of user data (e.g., payment information, two-dimensional avatars of users, three-dimensional avatars of users, and avatars with various avatar characteristics). Avatar characteristics include different ways to customize a look and feel of an avatar, such as different poses, facial features, clothing, and so forth.

The visual code scanner system **214** implements various functions for scanning visual codes with respect to the messaging system **100**. For example, visual codes may correspond to QR code, 1Barcode, Data Matrix, Aztec code, and the like. In one or more embodiments, the client device **102** may access image data (e.g., a still image, a video) that includes the visual code. For example, the image data may be captured by the client device **102** in real time (e.g., a live image feed from a camera sensor of the client device **102**). In another example, the image data is stored on or received by the client device **102**, based on prior image capture via the client device **102** or another device. The visual code scanner system **214** is configured to access the image data that includes the visual code. In embodiments where the image data is video, the visual code scanner system **214** is configured to analyze individual frames of the video or a combination of multiple frames of the video to detect and decode the visual code.

In one or more embodiments, the visual code scanner system **214** is configured to identify a custom graphic (e.g., a ghost icon) in the image and use the custom graphic as an alignment pattern for decoding data included in the visual code. For example, the visual code scanner system **214** extracts candidate shape features from the image data of the image. Subsequently, the visual code scanner system **214** determines if the candidate features meet certain rules and criteria to filter out irrelevant shape features or shape features that have a low probability of being the custom graphic. The visual code scanner system **214** then compares the candidate shape features that meet the shape feature criteria or rules with reference shape features of the custom graphic. In an example, the visual code scanner system **214** identifies the custom graphic based on a match between the candidate shape features and the reference shape feature (e.g., a match score that exceeds a threshold).

Subsequent to identifying the custom graphic, the visual code scanner system **214** uses the custom graphic as an alignment pattern for decoding. For instance, the visual code scanner system **214** extracts spatial attributes of the custom graphic in the image and compares the extracted spatial attributes to reference spatial attributes to determine an alignment of the custom graphic. The visual code scanner system **214** may then generate a transformed image of the image according to the alignment (e.g., a rotation or deskew). After generating the transformed image, the visual code scanner system **214** decodes the data encoded in a portion of the transformed image. For example, the dots of the visual code are transformed into data shown as ones for dots and zeros for non-dots, although this is merely an illustrative example and other schemes can be employed. In this way, visual code scanner system **214** uses the custom graphic included in the visual code as one or more functional patterns such as a finder pattern or an alignment pattern.

Based on decoding the data encoded in the image, the visual code scanner system **214** facilitates access to features and/or unique content of the messaging system **100**. The data encoded in a portion of the image can indicate a particular feature (e.g., action to perform) or include information to be used in conjunction with a particular feature. Example features include, but are not limited to: accessing/activating an augmented reality content item (e.g., unlocking a Lens), adding friends, linking to a website, connecting to particular content, and the like. For example, a unique visual code may be assigned to each augmented reality content item (e.g., upon generation of the augmented reality content item), friend, website and/or other content, to provide access thereto by decoding the corresponding unique visual code.

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

The database **126** includes message data stored within a message table **306**. This message data includes, for any particular one message, at least message sender data, message recipient (or receiver) data, and a payload. Further details regarding information that may be included in a message, and included within the message data stored in the message table **306** is described below with reference to FIG. **4**.

An entity table **308** stores entity data, and is linked (e.g., referentially) to an entity graph **304** and profile data **302**. Entities for which records are maintained within the entity table **308** may include individuals, corporate entities, organizations, objects, places, events, and so forth. Regardless of entity 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).

The entity graph **304** 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.

The profile data **302** stores multiple types of profile data about a particular entity. The profile data **302** may be selectively used and presented to other users of the messaging system **100**, based on privacy settings specified by a particular entity. Where the entity is an individual, the profile data **302** includes, for example, a user name, telephone number, address, settings (e.g., notification and privacy settings), as well as a user-selected avatar representation (or collection of such avatar representations). A particular user may then selectively include one or more of these avatar representations within the content of messages communicated via the messaging system **100**, and on map interfaces displayed by messaging clients **104** to other users. The collection of avatar representations may include "status avatars," which present a graphical representation of a status or activity that the user may select to communicate at a particular time.

Where the entity is a group, the profile data **302** for the group may similarly include one or more avatar representations associated with the group, in addition to the group name, members, and various settings (e.g., notifications) for the relevant group.

In one or more embodiments, the profile data **302** includes a respective visual code for each entity. For example, the visual code for a particular entity is scannable in order to access (e.g., add) that entity as a friend.

The database **126** also stores augmentation data, such as overlays or filters, in an augmentation table **310**. The aug-

mentation data is associated with and applied to videos (for which data is stored in a video table **314**) and images (for which data is stored in an image table **316**).

Filters, in one example, are overlays that are displayed as overlaid on an image or video during presentation to a recipient user. Filters may be of various types, including user-selected filters from a set of filters presented to a sending user by the messaging client **104** when the sending user is composing a message. Other types of filters include geolocation filters (also known as geo-filters), which may be presented to a sending user based on geographic location. For example, geolocation filters specific to a neighborhood or special location may be presented within a user interface by the messaging client **104**, based on geolocation information determined by a Global Positioning System (GPS) unit of the client device **102**.

Another type of filter is a data filter, which may be selectively presented to a sending user by the messaging client **104**, based on other inputs or information gathered by the client device **102** during the message creation process. Examples of data filters include current temperature at a specific location, a current speed at which a sending user is traveling, battery life for a client device **102**, or the current time.

Other augmentation data that may be stored within the augmentation table **310** (e.g., in conjunction with the image table **316** and/or the video table **314**) includes augmented reality content items (e.g., corresponding to applying Lenses or augmented reality experiences). An augmented reality content item may provide a real-time special effect and/or sound that may be added to an image or a video. Each of the augmented reality content items may be available for selection and activation by the messaging client **104**.

As described above, augmentation data includes augmented reality content items, overlays, image transformations, AR images, and similar terms refer to modifications that may be applied to image data (e.g., videos or images). This includes real-time modifications, which modify an image as it is captured using device sensors (e.g., one or multiple cameras) of a client device **102** and then displayed on a screen of the client device **102** with the modifications. This also includes modifications to stored content, such as video clips in a gallery that may be modified. For example, in a client device **102** with access to multiple augmented reality content items, a user can use a single video clip with multiple augmented reality content items to see how the different augmented reality content items will modify the stored clip. For example, multiple augmented reality content items that apply different pseudorandom movement models can be applied to the same content by selecting different augmented reality content items for the content. Similarly, real-time video capture may be used with an illustrated modification to show how video images currently being captured by sensors of a client device **102** would modify the captured data. Such data may simply be displayed on the screen and not stored in memory, or the content captured by the device sensors may be recorded and stored in memory with or without the modifications (or both). In some systems, a preview feature can show how different augmented reality content items will look within different windows in a display at the same time. This can, for example, enable multiple windows with different pseudorandom animations to be viewed on a display at the same time.

Data and various systems using augmented reality content items or other such transform systems to modify content using this data can thus involve detection of objects (e.g., faces, hands, bodies, cats, dogs, surfaces, objects, etc.), tracking of such objects as they leave, enter, and move around the field of view in video frames, and the modification or transformation of such objects as they are tracked. In various examples, different methods for achieving such transformations may be used. Some examples may involve generating a three-dimensional mesh model of the object or objects, and using transformations and animated textures of the model within the video to achieve the transformation. In other examples, tracking of points on an object may be used to place an image or texture (which may be two dimensional or three dimensional) at the tracked position. In still further examples, neural network analysis of video frames may be used to place images, models, or textures in content (e.g., images or frames of video). Augmented reality content items thus refer both to the images, models, and textures used to create transformations in content, as well as to additional modeling and analysis information needed to achieve such transformations with object detection, tracking, and placement.

Real-time video processing can be performed with any kind of video data (e.g., video streams, video files, etc.) saved in a memory of a computerized system of any kind. For example, a user can load video files and save them in a memory of a device, or can generate a video stream using sensors of the device. Additionally, any objects can be processed using a computer animation model, such as a human's face and parts of a human body, animals, or non-living things such as chairs, cars, or other objects.

In some examples, when a particular modification is selected along with content to be transformed, elements to be transformed are identified by the computing device, and then detected and tracked if they are present in the frames of the video. The elements of the object are modified according to the request for modification, thus transforming the frames of the video stream. Transformation of frames of a video stream can be performed by different methods for different kinds of transformation. For example, for transformations of frames mostly referring to changing forms of object's elements characteristic points for each element of an object are calculated (e.g., using an Active Shape Model (ASM) or other known methods). Then, a mesh based on the characteristic points is generated for each of the at least one element of the object. This mesh used in the following stage of tracking the elements of the object in the video stream. In the process of tracking, the mentioned mesh for each element is aligned with a position of each element. Then, additional points are generated on the mesh. A first set of first points is generated for each element based on a request for modification, and a set of second points is generated for each element based on the set of first points and the request for modification. Then, the frames of the video stream can be transformed by modifying the elements of the object on the basis of the sets of first and second points and the mesh. In such method, a background of the modified object can be changed or distorted as well by tracking and modifying the background.

In some examples, transformations changing some areas of an object using its elements can be performed by calculating characteristic points for each element of an object and generating a mesh based on the calculated characteristic points. Points are generated on the mesh, and then various areas based on the points are generated. The elements of the object are then tracked by aligning the area for each element with a position for each of the at least one element, and properties of the areas can be modified based on the request for modification, thus transforming the frames of the video stream. Depending on the specific request for modification

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

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

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

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

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

In some examples, a computer animation model to transform image data can be used by a system where a user may capture an image or video stream of the user (e.g., a selfie) using a client device **102** having a neural network operating as part of a messaging client **104** operating on the client device **102**. The transformation system operating within the messaging client **104** determines the presence of a face within the image or video stream and provides modification icons associated with a computer animation model to transform image data, or the computer animation model can be present as associated with an interface described herein. The modification icons include changes that may be the basis for modifying the user's face within the image or video stream as part of the modification operation. Once a modification icon is selected, the transform system initiates a process to convert the image of the user to reflect the selected modification icon (e.g., generate a smiling face on the user). A modified image or video stream may be presented in a graphical user interface displayed on the client device **102** as soon as the image or video stream is captured, and a specified modification is selected. The transformation system may implement a complex convolutional neural network on a portion of the image or video stream to generate and apply the selected modification. That is, the user may capture the image or video stream and be presented with a modified result in real-time or near real-time once a modification icon has been selected. Further, the modification may be persistent while the video stream is being captured, and the selected modification icon remains toggled. Machine taught neural networks may be used to enable such modifications.

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

In one or more embodiments, the augmentation table **310** stores a respective visual code for each augmented reality content item (or overlay, image transformation, AR image, and the like) that may be applied to image data. For example, the visual code for a particular augmented reality content item is scannable in order to access (e.g., activate, unlock) that augmented reality content item.

A story table **312** 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 **308**). A user may create a "personal story" in the form of a collection of content that has been created and sent/broadcast by that user. To this end, the user interface of the messaging client **104** may include an icon that is user-selectable to enable a sending user to add specific content to his or her personal story.

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

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 examples, a contribution to a location story may require a second degree of authentication to verify that the end user belongs to a specific organization or other entity (e.g., is a student on the university campus).

As mentioned above, the video table **314** stores video data that, in one example, is associated with messages for which records are maintained within the message table **306**. Similarly, the image table **316** stores image data associated with messages for which message data is stored in the entity table **308**. The entity table **308** may associate various augmentations from the augmentation table **310** with various images and videos stored in the image table **316** and the video table **314**.

FIG. **4** is a schematic diagram illustrating a structure of a message **400**, according to some examples, generated by a messaging client **104** for communication to a further messaging client **104** or the messaging server **118**. The content of a particular message **400** is used to populate the message table **306** stored within the database **126**, accessible by the messaging server **118**. 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 servers **114**. A message **400** is shown to include the following example components:

- message identifier **402**: a unique identifier that identifies the message **400**.
- 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**.
- message image payload **406**: image data, captured by a camera component of a client device **102** or retrieved from a memory component of a client device **102**, and that is included in the message **400**. Image data for a sent or received message **400** may be stored in the image table **316**.
- 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**. Video data for a sent or received message **400** may be stored in the video table **314**.
- message audio payload **410**: audio data, captured by a microphone or retrieved from a memory component of the client device **102**, and that is included in the message **400**.
- message augmentation data **412**: augmentation data (e.g., filters, stickers, or other annotations or enhancements) that represents augmentations to be applied to message image payload **406**, message video payload **408**, or message audio payload **410** of the message **400**. Augmentation data for a sent or received message **400** may be stored in the augmentation table **310**.
- 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 **104**.
- 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, 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**).
- message story identifier **418**: identifier values identifying one or more content collections (e.g., "stories" identified in the story table **312**) 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.
- 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.
- 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.
- 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.

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 **316**. Similarly, values within the message video payload **408** may point to data stored within a video table **314**, values stored within the message augmentations **412** may point to data stored in an augmentation table **310**, values stored within the message story identifier **418** may point to data stored in a story table **312**, 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 **308**.

FIG. **5** is an interaction diagram illustrating a process **500** for interacting with visual codes for accessing features within a messaging system, in accordance with some examples. For explanatory purposes, the process **500** is primarily described herein with reference to the messaging client **104** and the messaging server system **108** of FIG. **1**. However, one or more blocks (or operations) of the process **500** may be performed by one or more other components, and/or by other suitable devices. Further for explanatory purposes, the blocks (or operations) of the process **500** are described herein as occurring in serial, or linearly. However, multiple blocks (or operations) of the process **500** may occur in parallel or concurrently. In addition, the blocks (or operations) of the process **500** need not be performed in the order shown and/or one or more blocks (or operations) of the process **500** need not be performed and/or can be replaced by other operations. The process **500** may be terminated

when its operations are completed. In addition, the process **500** may correspond to a method, a procedure, an algorithm, etc.

In one or more embodiments, the messaging client **104** is running on a client device **102** associated with a user of messaging server system **108**. The user is associated with a user account of the messaging server system **108**. For example, the user is identified by the messaging server system **108** based on one or more unique identifiers (e.g., a messaging system identifier, email address and/or a device identifier) associated with the user accounts for the user. In some embodiments, the messaging server system **108** implements and/or works in conjunction with a social network server **124** which is configured to identify other users (e.g., friends) with which the user of the messaging client **104** has relationships.

As noted above, each of the visual code scanner system **214** and the augmentation system **208** may correspond to respective subsystems of the messaging system **100**, and may be supported on the client side by the messaging client **104** and/or on the server side by the application servers **114**. Thus, the operations described herein with respect to the visual code scanner system **214** and the augmentation system **208** may be implemented client side, server side and/or a combination of client side and server side.

As described herein, the messaging system **100** provides for a user to interact with visual codes, to access features of the messaging system **100**. In response to user selection of a visual code in a captured image, the messaging client **104** provides for updating a custom graphic within the visual code. The messaging client **104** animates movement of the custom graphic from the visual code to a carousel interface with user-selectable icons for respective augmented reality content, and further updates the carousel interface with an icon corresponding to the visual code. The code (e.g., assets) for effecting the corresponding augmented reality content item can be automatically downloaded and/or activated, thereby reducing the amount of interaction required from the user.

At block **502**, the messaging client **104** running on the messaging client **104** displays captured image data that includes a visual code with a custom graphic and markings. As noted above, the messaging client **104** is configured access image data (e.g., a still image, a video) that includes the visual code. The image data may be captured by the client device **102** in real time (e.g., a live image feed from a camera sensor of the client device **102**). For example, the messaging client **104** may activate a camera upon initiation of the messaging client **104**. The messaging client **104** may also include interface elements for activating the camera, and for switching between a front-facing and rear-facing camera.

As an alternative to real-time capture of image data, the image data may correspond to a prior image captured by the client device **102** or by another device. For example, the prior image may have been sent as a message to the user of the messaging client **104**, or the user may have selected to view/download the prior image via the messaging client **104**.

The captured image data may correspond to an object, such as a poster, a computer display, an article of clothing, or a piece of printed paper. The visual code may be printed or otherwise appear on the object. In one or more embodiments, the visual code is square-shaped with rounded edges.

In one or more embodiments, the visual code includes a custom graphic and marking. The custom graphic can be employed as a finder pattern used for identification and recognition of a visual code, or an alignment pattern used to facilitate decoding. In the below examples of FIGS. **6-10**, the custom graphic is depicted as a ghost icon. However, other example custom graphics may be used, and can include designs with multiple paths, multiple polygons, multiple aesthetic elements, or other design features.

Thus, the messaging client **104**, in conjunction with the visual code scanner system **214**, identifies the custom graphic (e.g., a ghost icon) in the image data and uses the custom graphic as an alignment pattern for decoding data included in the visual code. As noted above, the visual code scanner system **214** extracts candidate shape features from the image data, applies rules and criteria to filter out irrelevant shape features, and performs comparisons with a reference shape feature based on a match score that exceeds a threshold. The visual code scanner system **214** further uses the custom graphic as an alignment pattern for decoding, and generates a transformed image of the image data according to the alignment.

As noted above, the visual code further includes markings (e.g., dots) arranged in a unique configuration which is decodable to access a particular feature of the messaging system **100**. In the example of FIG. **5**, the feature corresponds to an augmented reality content item. However, the features as described herein are is not limited to augmented reality content items, and may instead relate to other features such as adding a friend, linking to a website, connecting to particular content, and the like.

Thus, at block **502**, the messaging client **104** displays a visual code included within captured image data. The visual code includes a custom graphic (e.g., which may be transformed for display based on alignment) and markings. At block **504**, the user of the client device **102** selects (e.g., via a tap or press gesture) the visual code as displayed on the messaging client **104**. In response to such user selection, the visual code scanner system **214** decodes the markings of the visual code. By way of non-limiting example, the markings (e.g., dots) of the visual code are decoded into data shown as ones for dots and zeros for non-dots, to identify a particular feature (e.g., augmented reality content item).

At block **506**, the messaging client **104** displays an updated custom graphic. For example, the messaging client **104** causes the custom graphic to be modified to indicate the user selection received at block **504**. By way of non-limiting example, and as discussed below with respect to FIGS. **7-8**, the original custom graphic of a ghost icon with two open eyes is updated to wink. Updating the custom graphic may correspond with replacing the original custom graphic (e.g., the non-winking ghost icon) with a static image (e.g., the winking ghost icon). Alternatively, updating the custom graphic may correspond with animating the custom graphic (e.g., animating the ghost icon to wink or to perform another predefined action).

In one example, the updated custom graphic is a default graphic (e.g., a default static image or a default animation) that is predefined (e.g., by a system administrator). In another example, the updated custom graphic may be set based on the code (e.g., assets) for the corresponding augmented reality content item. As described herein, the visual code corresponds with an augmented reality content item that has code associated therewith. The code includes the stored assets for implementing the augmented reality (e.g., visual and/or audio effects) corresponding to the augmented reality content item. In addition, the code may include a specific icon for the augmented reality content item, where the specific icon is usable within a carousel interface.

In this regard, as discussed further below with respect to FIG. **9**, the messaging client **104** at block **508** provides for animating the custom graphic as moving from within the display area of the visual code to a carousel interface. In one or more embodiments, the carousel interface provided by the messaging client **104** allows the user to cycle through and/or select a different augmented reality content item (e.g., Lens) to apply with respect to captured image data. Each of the available augmented reality content items is represented by a specific icon (e.g., a circular icon) which is user-selectable for switching to the respective augmented reality content item. As noted above, each of the augmented reality content items has corresponding code which implements the effect (e.g., video and/or audio effect) and the specific icon for that augmented reality content item.

An icon corresponding to a currently active augmented reality content item may be displayed in a different manner relative to the remaining icons. For example, the icon for the active augmented reality content item is displayed as centered, larger than and/or highlighted relative to the remaining icons in the carousel interface. In one or more embodiments, the remaining icons are displayed in a ranked order, for example, based on predefined rankings (e.g., one or more of popularity, personal history, promotions, and the like).

In addition, the messaging client **104** at block **510** updates the carousel interface to include an additional icon for the augmented reality content item corresponding to the visual code. For example, the messaging client **104** provides an animation which depicts the additional icon as being inserted between two existing icons of the carousel interface.

In one or more embodiments, the messaging client **104** may provide for updating the custom graphic (e.g., block **506**), moving the visual code toward the carousel interface (e.g., block **508**), and inserting the additional icon into the carousel interface (e.g., block **510**) to be a smooth, animated transition. For example, the messaging client **104** causes the custom graphic to move towards the carousel interface, and to transform (e.g., scale, rotate, and the like) while moving. In addition, the messaging client **104** provides for animating the existing icons in the carousel interface to make space for the additional icon, and for smoothly changing the custom graphic from the one that is visible on the code (e.g., the original ghost icon) to the specific icon created for the augmented reality content item.

At operation **512**, the messaging client **104** sends, to the messaging server system **108**, a request for the code of the augmented reality content item corresponding to the visual code. In response, the messaging server system **108** sends, to the messaging client **104**, the code of the augmented reality content item corresponding to the visual code (operation **514**).

As noted above, the code may include the assets for implementing/effecting the augmented reality content item. In one or more embodiments, the code is stored in the augmentation table **310** of the database **126**. The request of operation **512** may include an identifier for the augmented reality content item, where the identifier is based on the decoding of the markings as described above.

In response to receiving the code, the messaging client **104** provides for activating the augmented reality content item with respect to the captured image data (block **516**). As noted above, activating the augmented reality content item includes presenting the visual and/or audio effects associated with the augmented reality content item in conjunction with the captured image data.

As discussed further below with respect to FIGS. **10**A-**10**C, the messaging client **104** may implement different options for downloading the code for the augmented reality content item, and/or for activating the augmented reality content item. Selection of the option may be determined by the user or a system administrator, based on appropriate interfaces provided by the messaging system **100**.

In a first option, the messaging client **104** requests the code (e.g., per operation **512**), and downloads the code (e.g., per operation **514**) corresponding to the augmented reality content item in response to the user input selecting the visual code at block **504**. In addition, the messaging client **104** activates the augmented reality content item (e.g., per block **516**) in response to second user input (not shown in FIG. **5**) that selects the additional icon that was inserted in the carousel interface (e.g., per block **510**).

In a second option, the messaging client **104** requests the code (e.g., per operation **512**), downloads the code (e.g., per operation **514**) corresponding to the augmented reality content item, and activates the augmented reality content item (e.g., per block **516**), all in response to the user input selecting the code at block **504**. Thus, the second option differs from the first option in that the second option does not require the second user input to activate the augmented reality content item.

In a third option, the messaging client **104** requests the code (e.g., per operation **512**), and downloads the code (e.g., per operation **514**) corresponding to the augmented reality content item in response to second user input (not shown in FIG. **5**) that selects the additional icon inserted in the carousel interface (e.g., per block **510**). For example, the additional icon is initially grayed out when inserted into the carousel interface. User selection of the grayed out icon causes the messaging client **104** to request and download the code corresponding to the augmented reality content item. The messaging client **104** may provide an animation to show download progress. For example, the animation may progressively remove the gray mask for the icon to indicate download progress and completion.

While the example of FIG. **5** relates to interaction with respect to visual codes for augmented reality content items, the messaging system **100** is not limited to such. The messaging system **100** may provide for similar interaction with respect to visual codes for other features provide by the messaging system **100**. Examples of other such features include, but are not limited to, adding a friend, linking to a website, connecting to particular content, and the like.

Thus, the messaging system **100** as described herein provides for interacting with visual codes for accessing features of the messaging system **100**. In other systems, after selecting a visual code, the user is presented with a separate message box (e.g., overlay) that prompts the user to download and/or activate the feature. This prompt may be distracting, as it switches the user's attention away from the current interface related to the feature. The messaging system **100** provides for a more seamless transition in downloading and/or activating the feature. Without providing for the visual code interactions as described herein, end users may be provided with additional prompts that can be distracting and interfere with user interface flow. As such, the messaging system **100** saves time for the user, and/or reduces computational resources/processing power associated with additional prompts and associated delays. Moreover, the visual code interactions as described herein may be more engaging for users of the messaging system **100**.

FIG. **6** illustrates a user interface **602** in which a visual code is displayed within captured image data, in accordance with some example embodiments. As shown in the example of FIG. **6**, the user interface **602** includes a visual code **604**,

markings **606**, a custom graphic **608**, captured image data **610** and a carousel interface **612**.

In one or more embodiments, the user interface **602** is provided by the messaging client **104**. The user interface **602** provides for displaying augmented reality content in association with the captured image data **610**. As noted above, the captured image data **610** may correspond to image data captured by a front-facing camera or a rear-facing camera of the client device **102**. Alternatively, the image data may correspond to a prior image captured via the client device **102** or another device.

The augmented reality content displayed in association with the captured image data **610** is based on a currently-selected augmented reality content item (e.g., Lens). In this regard, the carousel interface **612** includes a set of icons, each of which is user-selectable (e.g., by a tap/press gesture) to select/activate a respective augmented reality content item. The carousel interface **612** allows the user to cycle through the icons (e.g., via swipe gestures), and to select a particular augmented reality content item to apply with respect to the captured image data **610**.

While not shown in the example of FIG. **6**, each icon may have a respective graphic to indicate its respective augmented reality content item (e.g., a hair graphic to indicate a hair effect, a smiling graphic to indicate a smile effect, and the like). The icon for the active augmented reality content item may be centered, larger than and/or highlighted relative to the remaining icons in the carousel interface **612**.

In the example of FIG. **6**, the captured image data **610** includes a visual code **604**. For example, the captured image data **610** may correspond to an object (e.g., a poster, a computer display, an article of clothing, or a piece of printed paper), and the visual code **604** may appear on the object. The visual code includes the custom graphic **608** and markings **606**.

As noted above, the custom graphic **608** may be employed as a finder pattern used for identification and recognition of the visual code **604**. In the example of FIG. **6**, the custom graphic **608** is depicted as a ghost icon. However, other custom graphics may be used.

The markings **606** of the visual code **604** are arranged in a unique configuration which is decodable to access a particular feature of the messaging system **100**. In the example of FIG. **6**, the feature corresponds to an augmented reality content item. User selection of the visual code **604** provides the user with access to the corresponding augmented reality content item.

FIG. **7** illustrates the user interface **602** in a case where the user selects the visual code **604** to activate an augmented reality content item, in accordance with some example embodiments. In the example of FIG. **7**, the user selection is depicted as touch input **702** (e.g., a tap or press gesture) performed with respect to the visual code **604**. In response to the touch input **702**, the visual code scanner system **214** decodes the markings of the visual code (e.g., to identify the corresponding augmented reality content item), and proceeds with presenting animations discussed below with respect to FIGS. **8**, **9** and **10**A-**10**C.

FIG. **8** illustrates the user interface **602** in which the custom graphic **608** of the visual code **604** is updated in response to user selection (e.g., touch input **702**), in accordance with some example embodiments. The update to the custom graphic **608** visually indicates, to the user of the client device **102**, receipt of the touch input **702**. As shown in the example of FIG. **8**, the original custom graphic **608** of the ghost icon is updated to wink. Updating the custom graphic **608** may correspond with replacing the original custom graphic (e.g., the non-winking ghost icon) with the predefined static image (e.g., the winking ghost icon). In one or more embodiments, updating the custom graphic **608** corresponds with animating the custom graphic (e.g., animating the ghost icon to wink or to perform another predefined action).

FIG. **9** illustrates the user interface **602** in which the updated custom graphic **608** of FIG. **8** is animated to move toward and be inserted within the carousel interface **612**, in accordance with some example embodiments. In the example of FIG. **9**, the animation **902** represents movement of the updated custom graphic **608** from the display area of the visual code **604** to the display area of the carousel interface **612**. The depicted arrows of the animation **902** are for illustrative purposes, to indicate such movement.

With respect to FIG. **10**A, the messaging client **104** updates the carousel interface **612** to include an additional icon **1002** for the augmented reality content item corresponding to the visual code **604**. For example, the messaging client **104** provides an animation which depicts the additional icon **1002** as being inserted between two existing icons of the carousel interface **612**. As noted above, the carousel interface **612** allows the user to cycle through augmented reality content items based on a predefined user gesture. While the example of FIG. **10**A illustrates four icons, the messaging client **104** is configured to present additional icons in response the predefined user gesture (e.g., a swipe gesture). For example, in response to a swipe gesture, the carousel interface **612** presents icons that are not presently displayed, with the icons being ranked as described above.

FIGS. **10**A-**10**C illustrate different manners for downloading and/or activating the augmented reality content item corresponding to the visual code **604**, in accordance with some example embodiments. As noted above, the messaging client **104** may implement different options for downloading the code (e.g., assets) for the augmented reality content item, and/or for activating the augmented reality content item.

FIG. **10**A illustrates both a first and second option, as discussed above with respect to FIG. **5**. In the first option, the messaging client **104** requests the code (e.g., assets) for the augmented reality content item, and downloads the code in response to the first user input selecting the code (e.g., block **504** of FIG. **5**). In addition, the messaging client **104** activates the augmented reality content item in response to second user input that selects the icon **1002** inserted in the carousel interface **612**.

In the second option, the messaging client **104** requests the code (e.g., assets) for the augmented reality content item, downloads the code in response to the first user input selecting the code, and activates the augmented reality content item in response to the first user input selecting the code (e.g., block **504** of FIG. **5**). Thus, the second option differs from the first option in that the second option does not require the second user input that selects the icon **1002**.

FIG. **10**B illustrates a third option as discussed above with respect to FIG. **5**. After inserting the icon **1002** into the carousel interface **612**, the messaging client **104** receives second user input that selects the icon **1002**. In response to receiving the second user input, the messaging client **104** requests the code (e.g., assets) corresponding to the augmented reality content item, and downloads the code.

As shown in the example of FIG. **10**B, the icon **1002** is initially grayed out when inserted into the carousel interface **612**. User selection of the grayed out icon causes the messaging client **104** to request and download the code corresponding to the augmented reality content item.

FIG. **10**C depicts an expanded view of the icon **1002**, to illustrate the messaging client **104** providing an animation showing a download progress for the augmented reality content item. For example, the animation progressively removes the gray mask for the icon **1002** to indicate download progress and completion.

FIG. **11** is a flowchart illustrating a process **1100** for interacting with visual codes for accessing features within a messaging system, in accordance with some examples. For explanatory purposes, the process **1100** is primarily described herein with reference to the messaging client **104** of FIG. **1**. However, one or more blocks (or operations) of the process **1100** may be performed by one or more other components, and/or by other suitable devices. Further for explanatory purposes, the blocks (or operations) of the process **1100** are described herein as occurring in serial, or linearly. However, multiple blocks (or operations) of the process **1100** may occur in parallel or concurrently. In addition, the blocks (or operations) of the process **1100** need not be performed in the order shown and/or one or more blocks (or operations) of the process **1100** need not be performed and/or can be replaced by other operations. The process **1100** may be terminated when its operations are completed. In addition, the process **1100** may correspond to a method, a procedure, an algorithm, etc.

The messaging client **104** displays captured image data comprising a visual code, the visual code including a custom graphic and being decodable to access a first feature of the messaging application (block **1102**). The visual code may include a unique configuration of markings which is decodable to access to the first feature. The messaging client **104** receives user input selecting the visual code (block **1104**).

The messaging client **104** displays an updated version of the custom graphic (block **1106**). The updated version of the custom graphic may correspond to a predefined static image. Alternatively, the updated version of the custom graphic may correspond to a predefined animation.

The messaging client **104** provides an animation which depicts the updated version of the custom graphic as moving from the visual code to an interface element comprising a group of icons, each icon within the group of icons being user-selectable to access a respective second feature of the messaging application (block **1108**).

The interface element may be a carousel interface for navigating the group of icons. The first feature and each of the respective second features may correspond to a respective augmented reality content item for applying to the captured image data.

The messaging client **104** may download, from the messaging server system **108**, code for a first augmented reality content item corresponding to the first feature. The messaging client **104** may activate, based on downloading the code, the first augmented reality content item with respect to the captured image data.

Downloading the code may be in response to second user input selecting the additional icon within the group of icons. Alternatively, downloading the code may be in response to receiving the user input (per block **1104**), and activating the first augmented reality content item may be in response to receiving second user input selecting the additional icon within the group of icons.

The messaging client **104** updates the group of icons to include an additional icon which is user-selectable to access the first feature of the messaging application (block **1110**).

FIG. **12** is a schematic diagram illustrating an access-limiting process **1200**, in terms of which access to content (e.g., an ephemeral message **1202**, and associated multimedia payload of data) or a content collection (e.g., an ephemeral message group **1204**) may be time-limited (e.g., made ephemeral).

An ephemeral message **1202** is shown to be associated with a message duration parameter **1206**, the value of which determines an amount of time that the ephemeral message **1202** will be displayed to a receiving user of the ephemeral message **1202** by the messaging client **104**. In one example, an ephemeral message **1202** is viewable by a receiving user for up to a maximum of 10 seconds, depending on the amount of time that the sending user specifies using the message duration parameter **1206**.

The message duration parameter **1206** and the message receiver identifier **424** are shown to be inputs to a message timer **1210**, which is responsible for determining the amount of time that the ephemeral message **1202** is shown to a particular receiving user identified by the message receiver identifier **424**. In particular, the ephemeral message **1202** will only be shown to the relevant receiving user for a time period determined by the value of the message duration parameter **1206**. The message timer **1210** is shown to provide output to a more generalized ephemeral timer system **202**, which is responsible for the overall timing of display of content (e.g., an ephemeral message **1202**) to a receiving user.

The ephemeral message **1202** is shown in FIG. **12** to be included within an ephemeral message group **1204** (e.g., a collection of messages in a personal story, or an event story). The ephemeral message group **1204** has an associated group duration parameter **1208**, a value of which determines a time duration for which the ephemeral message group **1204** is presented and accessible to users of the messaging system **100**. The group duration parameter **1208**, for example, may be the duration of a music concert, where the ephemeral message group **1204** is a collection of content pertaining to that concert. Alternatively, a user (either the owning user or a curator user) may specify the value for the group duration parameter **1208** when performing the setup and creation of the ephemeral message group **1204**.

Additionally, each ephemeral message **1202** within the ephemeral message group **1204** has an associated group participation parameter **1212**, a value of which determines the duration of time for which the ephemeral message **1202** will be accessible within the context of the ephemeral message group **1204**. Accordingly, a particular ephemeral message group **1204** may "expire" and become inaccessible within the context of the ephemeral message group **1204**, prior to the ephemeral message group **1204** itself expiring in terms of the group duration parameter **1208**. The group duration parameter **1208**, group participation parameter **1212**, and message receiver identifier **424** each provide input to a group timer **1214**, which operationally determines, firstly, whether a particular ephemeral message **1202** of the ephemeral message group **1204** will be displayed to a particular receiving user and, if so, for how long. Note that the ephemeral message group **1204** is also aware of the identity of the particular receiving user as a result of the message receiver identifier **424**.

Accordingly, the group timer **1214** operationally controls the overall lifespan of an associated ephemeral message group **1204**, as well as an individual ephemeral message **1202** included in the ephemeral message group **1204**. In one example, each and every ephemeral message **1202** within the ephemeral message group **1204** remains viewable and accessible for a time period specified by the group duration parameter **1208**. In a further example, a certain ephemeral message **1202** may expire, within the context of ephemeral

message group **1204**, based on a group participation parameter **1212**. Note that a message duration parameter **1206** may still determine the duration of time for which a particular ephemeral message **1202** is displayed to a receiving user, even within the context of the ephemeral message group **1204**. Accordingly, the message duration parameter **1206** determines the duration of time that a particular ephemeral message **1202** is displayed to a receiving user, regardless of whether the receiving user is viewing that ephemeral message **1202** inside or outside the context of an ephemeral message group **1204**.

The ephemeral timer system **202** may furthermore operationally remove a particular ephemeral message **1202** from the ephemeral message group **1204** based on a determination that it has exceeded an associated group participation parameter **1212**. For example, when a sending user has established a group participation parameter **1212** of 24 hours from posting, the ephemeral timer system **202** will remove the relevant ephemeral message **1202** from the ephemeral message group **1204** after the specified 24 hours. The ephemeral timer system **202** also operates to remove an ephemeral message group **1204** when either the group participation parameter **1212** for each and every ephemeral message **1202** within the ephemeral message group **1204** has expired, or when the ephemeral message group **1204** itself has expired in terms of the group duration parameter **1208**.

In certain use cases, a creator of a particular ephemeral message group **1204** may specify an indefinite group duration parameter **1208**. In this case, the expiration of the group participation parameter **1212** for the last remaining ephemeral message **1202** within the ephemeral message group **1204** will determine when the ephemeral message group **1204** itself expires. In this case, a new ephemeral message **1202**, added to the ephemeral message group **1204**, with a new group participation parameter **1212**, effectively extends the life of an ephemeral message group **1204** to equal the value of the group participation parameter **1212**.

Responsive to the ephemeral timer system **202** determining that an ephemeral message group **1204** has expired (e.g., is no longer accessible), the ephemeral timer system **202** communicates with the messaging system **100** (and, for example, specifically the messaging client **104**) to cause an indicium (e.g., an icon) associated with the relevant ephemeral message group **1204** to no longer be displayed within a user interface of the messaging client **104**. Similarly, when the ephemeral timer system **202** determines that the message duration parameter **1206** for a particular ephemeral message **1202** has expired, the ephemeral timer system **202** causes the messaging client **104** to no longer display an indicium (e.g., an icon or textual identification) associated with the ephemeral message **1202**.

FIG. **13** is a diagrammatic representation of the machine **1300** within which instructions **1310** (e.g., software, a program, an application, an applet, an app, or other executable code) for causing the machine **1300** to perform any one or more of the methodologies discussed herein may be executed. For example, the instructions **1310** may cause the machine **1300** to execute any one or more of the methods described herein. The instructions **1310** transform the general, non-programmed machine **1300** into a particular machine **1300** programmed to carry out the described and illustrated functions in the manner described. The machine **1300** may operate as a standalone device or may be coupled (e.g., networked) to other machines. In a networked deployment, the machine **1300** 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 **1300** may comprise, but not be limited to, a server computer, a client computer, a personal computer (PC), a tablet computer, a laptop computer, a netbook, a set-top box (STB), a personal digital assistant (PDA), an entertainment media system, a cellular telephone, a smartphone, a mobile device, a wearable device (e.g., a smartwatch), a smart home device (e.g., a smart appliance), other smart devices, a web appliance, a network router, a network switch, a network bridge, or any machine capable of executing the instructions **1310**, sequentially or otherwise, that specify actions to be taken by the machine **1300**. Further, while only a single machine **1300** is illustrated, the term "machine" shall also be taken to include a collection of machines that individually or jointly execute the instructions **1310** to perform any one or more of the methodologies discussed herein. The machine **1300**, for example, may comprise the client device **102** or any one of a number of server devices forming part of the messaging server system **108**. In some examples, the machine **1300** may also comprise both client and server systems, with certain operations of a particular method or algorithm being performed on the server-side and with certain operations of the particular method or algorithm being performed on the client-side.

The machine **1300** may include processors **1304**, memory **1306**, and input/output I/O components **1302**, which may be configured to communicate with each other via a bus **1340**. In an example, the processors **1304** (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 **1308** and a processor **1312** that execute the instructions **1310**. The term "processor" is intended to include multi-core processors that may comprise two or more independent processors (sometimes referred to as "cores") that may execute instructions contemporaneously. Although FIG. **13** shows multiple processors **1304**, the machine **1300** may include a single processor with a single-core, a single processor with multiple cores (e.g., a multi-core processor), multiple processors with a single core, multiple processors with multiples cores, or any combination thereof.

The memory **1306** includes a main memory **1314**, a static memory **1316**, and a storage unit **1318**, both accessible to the processors **1304** via the bus **1340**. The main memory **1306**, the static memory **1316**, and storage unit **1318** store the instructions **1310** embodying any one or more of the methodologies or functions described herein. The instructions **1310** may also reside, completely or partially, within the main memory **1314**, within the static memory **1316**, within machine-readable medium **1320** within the storage unit **1318**, within at least one of the processors **1304** (e.g., within the Processor's cache memory), or any suitable combination thereof, during execution thereof by the machine **1300**.

The I/O components **1302** 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 **1302** that are included in a particular machine will depend on the type of machine. For example, portable machines such as mobile phones may include a touch input device or other such input mechanisms, while a headless server machine will likely not include such a touch input device. It will be appreciated that the I/O components **1302** may include many

other components that are not shown in FIG. **13**. In various examples, the I/O components **1302** may include user output components **1326** and user input components **1328**. The user output components **1326** may include visual components (e.g., a display such as a plasma display panel (PDP), a light-emitting diode (LED) display, a liquid crystal display (LCD), a projector, or a cathode ray tube (CRT)), acoustic components (e.g., speakers), haptic components (e.g., a vibratory motor, resistance mechanisms), other signal generators, and so forth. The user input components **1328** may include alphanumeric input components (e.g., a keyboard, a touch screen configured to receive alphanumeric input, a photo-optical keyboard, or other alphanumeric input components), point-based input components (e.g., a mouse, a touchpad, a trackball, a joystick, a motion sensor, or another pointing instrument), tactile input components (e.g., a physical button, a touch screen that provides location and force of touches or touch gestures, or other tactile input components), audio input components (e.g., a microphone), and the like.

In further examples, the I/O components **1302** may include biometric components **1330**, motion components **1332**, environmental components **1334**, or position components **1336**, among a wide array of other components. For example, the biometric components **1330** 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 **1332** include acceleration sensor components (e.g., accelerometer), gravitation sensor components, rotation sensor components (e.g., gyroscope).

The environmental components **1334** include, for example, one or cameras (with still image/photograph and video capabilities), illumination sensor components (e.g., photometer), temperature sensor components (e.g., one or more thermometers that detect ambient temperature), humidity sensor components, pressure sensor components (e.g., barometer), acoustic sensor components (e.g., one or more microphones that detect background noise), proximity sensor components (e.g., infrared sensors that detect nearby objects), gas sensors (e.g., gas detection sensors to detection concentrations of hazardous gases for safety or to measure pollutants in the atmosphere), or other components that may provide indications, measurements, or signals corresponding to a surrounding physical environment.

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

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

The position components **1336** 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.

Communication may be implemented using a wide variety of technologies. The I/O components **1302** further include communication components **1338** operable to couple the machine **1300** to a network **1322** or devices **1324** via respective coupling or connections. For example, the communication components **1338** may include a network interface Component or another suitable device to interface with the network **1322**. In further examples, the communication components **1338** 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 **1324** may be another machine or any of a wide variety of peripheral devices (e.g., a peripheral device coupled via a USB).

Moreover, the communication components **1338** may detect identifiers or include components operable to detect identifiers. For example, the communication components **1338** 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 **1338**, such as location via Internet Protocol (IP) geolocation, location via Wi-Fi® signal triangulation, location via detecting an NFC beacon signal that may indicate a particular location, and so forth.

The various memories (e.g., main memory **1314**, static memory **1316**, and memory of the processors **1304**) and storage unit **1318** may store one or more sets of instructions and data structures (e.g., software) embodying or used by any one or more of the methodologies or functions described herein. These instructions (e.g., the instructions **1310**), when executed by processors **1304**, cause various operations to implement the disclosed examples.

The instructions **1310** may be transmitted or received over the network **1322**, using a transmission medium, via a network interface device (e.g., a network interface component included in the communication components **1338**) and using any one of several well-known transfer protocols (e.g., hypertext transfer protocol (HTTP)). Similarly, the instructions **1310** may be transmitted or received using a transmission medium via a coupling (e.g., a peer-to-peer coupling) to the devices **1324**.

FIG. **14** is a block diagram **1400** illustrating a software architecture **1404**, which can be installed on any one or more of the devices described herein. The software architecture **1404** is supported by hardware such as a machine **1402** that includes processors **1420**, memory **1426**, and I/O components **1438**. In this example, the software architecture **1404** can be conceptualized as a stack of layers, where each layer provides a particular functionality. The software architecture **1404** includes layers such as an operating system **1412**,

libraries **1410**, frameworks **1408**, and applications **1406**. Operationally, the applications **1406** invoke API calls **1450** through the software stack and receive messages **1452** in response to the API calls **1450**.

The operating system **1412** manages hardware resources and provides common services. The operating system **1412** includes, for example, a kernel **1414**, services **1416**, and drivers **1422**. The kernel **1414** acts as an abstraction layer between the hardware and the other software layers. For example, the kernel **1414** provides memory management, processor management (e.g., scheduling), component management, networking, and security settings, among other functionality. The services **1416** can provide other common services for the other software layers. The drivers **1422** are responsible for controlling or interfacing with the underlying hardware. For instance, the drivers **1422** can include display drivers, camera drivers, BLUETOOTH® or BLUETOOTH® Low Energy drivers, flash memory drivers, serial communication drivers (e.g., USB drivers), WI-FI® drivers, audio drivers, power management drivers, and so forth.

The libraries **1410** provide a common low-level infrastructure used by the applications **1406**. The libraries **1410** can include system libraries **1418** (e.g., C standard library) that provide functions such as memory allocation functions, string manipulation functions, mathematic functions, and the like. In addition, the libraries **1410** can include API libraries **1424** such as media libraries (e.g., libraries to support presentation and manipulation of various media formats such as Moving Picture Experts Group-4 (MPEG4), Advanced Video Coding (H.264 or AVC), Moving Picture Experts Group Layer-3 (MP3), Advanced Audio Coding (AAC), Adaptive Multi-Rate (AMR) audio codec, Joint Photographic Experts Group (JPEG or JPG), or Portable Network Graphics (PNG)), graphics libraries (e.g., an OpenGL framework used to render in two dimensions (2D) and three dimensions (3D) in a graphic content on a display), database libraries (e.g., SQLite to provide various relational database functions), web libraries (e.g., WebKit to provide web browsing functionality), and the like. The libraries **1410** can also include a wide variety of other libraries **1428** to provide many other APIs to the applications **1406**.

The frameworks **1408** provide a common high-level infrastructure that is used by the applications **1406**. For example, the frameworks **1408** provide various graphical user interface (GUI) functions, high-level resource management, and high-level location services. The frameworks **1408** can provide a broad spectrum of other APIs that can be used by the applications **1406**, some of which may be specific to a particular operating system or platform.

In an example, the applications **1406** may include a home application **1436**, a contacts application **1430**, a browser application **1432**, a book reader application **1434**, a location application **1442**, a media application **1444**, a messaging application **1446**, a game application **1448**, and a broad assortment of other applications such as a third-party application **1440**. The applications **1406** are programs that execute functions defined in the programs. Various programming languages can be employed to create one or more of the applications **1406**, structured in a variety of manners, such as object-oriented programming languages (e.g., Objective-C, Java, or C++) or procedural programming languages (e.g., C or assembly language). In a specific example, the third-party application **1440** (e.g., an application developed using the ANDROID™ or IOS™ software development kit (SDK) by an entity other than the vendor of the particular platform) may be mobile software running on a mobile operating system such as IOS™, ANDROID™, WINDOWS® Phone, or another mobile operating system. In this example, the third-party application **1440** can invoke the API calls **1450** provided by the operating system **1412** to facilitate functionality described herein.

"Carrier signal" 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 media to facilitate communication of such instructions. Instructions may be transmitted or received over a network using a transmission medium via a network interface device.

"Client device" 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, portable digital assistants (PDAs), smartphones, tablets, ultrabooks, 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.

"Communication network" 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 types of cellular or wireless coupling. In this example, the coupling may implement any of a variety of types of data transfer technology, such as Single Carrier Radio Transmission Technology (1×RTT), Evolution-Data Optimized (EVDO) technology, General Packet Radio Service (GPRS) technology, Enhanced Data rates for GSM Evolution (EDGE) technology, third Generation Partnership Project (3GPP) including 3G, fourth generation wireless (4G) networks, Universal Mobile Telecommunications System (UMTS), High Speed Packet Access (HSPA), Worldwide Interoperability for Microwave Access (WiMAX), Long Term Evolution (LTE) standard, others defined by various standard-setting organizations, other long-range protocols, or other data transfer technology.

"Component" 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 examples, one or more computer systems (e.g., a standalone computer system, a client

computer system, or a server computer system) or one or more hardware 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 examples in which hardware components are temporarily configured (e.g., programmed), each of the hardware components need not be configured or instantiated at any one instance in time. For example, where a hardware component comprises a general-purpose processor configured by software to become a special-purpose processor, the general-purpose processor may be configured as respectively different special-purpose processors (e.g., comprising different hardware components) at different times. Software accordingly configures a particular processor or processors, for example, to constitute a particular hardware component at one instance of time and to constitute a different hardware component at a different instance of time. Hardware components can provide information to, and receive information from, other hardware components. Accordingly, the described hardware components may be regarded as being communicatively coupled. Where multiple hardware components exist contemporaneously, communications may be achieved through signal transmission (e.g., over appropriate circuits and buses) between or among two or more of the hardware components. In examples in which multiple hardware components are configured or instantiated at different times, communications between such hardware components may be achieved, for example, through the storage and retrieval of information in memory structures to which the multiple hardware components have access. For example, one hardware component may perform an operation and store the output of that operation in a memory device to which it is communicatively coupled. A further hardware component may then, at a later time, access the memory device to retrieve and process the stored output. 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 examples, the processors or processor-implemented components may be located in a single geographic location (e.g., within a home environment, an office environment, or a server farm). In other examples, the processors or processor-implemented components may be distributed across a number of geographic locations.

"Computer-readable storage medium" refers to both machine-storage media and transmission media. Thus, the terms include both storage devices/media and carrier waves/modulated data signals. The terms "machine-readable medium," "computer-readable medium" and "device-readable medium" mean the same thing and may be used interchangeably in this disclosure.

"Ephemeral message" 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.

"Machine storage medium" refers to a single or multiple storage devices and media (e.g., a centralized or distributed database, and associated caches and servers) that store executable instructions, routines and data. The term shall accordingly be taken to include, but not be limited to, solid-state memories, and optical and magnetic media, including memory internal or external to processors. Specific examples of machine-storage media, computer-storage media and device-storage media include non-volatile memory, including by way of example semiconductor memory devices, e.g., erasable programmable read-only memory (EPROM), electrically erasable programmable read-only memory (EEPROM), FPGA, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks The terms "machine-storage medium," "device-storage medium," "computer-storage medium" mean the same thing and may be used interchangeably in this disclosure. The terms "machine-storage media," "computer-storage media," and "device-storage media" specifically exclude carrier waves, modulated data signals, and other such media, at least some of which are covered under the term "signal medium."

"Non-transitory computer-readable storage medium" refers to a tangible medium that is capable of storing, encoding, or carrying the instructions for execution by a machine.

"Signal medium" refers to any intangible medium that is capable of storing, encoding, or carrying the instructions for execution by a machine and includes digital or analog communications signals or other intangible media to facilitate communication of software or data. The term "signal medium" shall be taken to include any form of a modulated data signal, carrier wave, and so forth. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a matter as to encode information in the signal. The terms "transmission medium" and "signal medium" mean the same thing and may be used interchangeably in this disclosure.

What is claimed is:

1. A method, comprising:

displaying, by an application running on a client device, captured image data comprising a visual code, the visual code including a custom graphic and being decodable to access a first feature of the application;

receiving, by the application, user input selecting the visual code; and

in response to receiving the user input,

updating a group of icons, each icon within the group of icons being user-selectable to access a respective second feature of the application, to include an additional icon which is user-selectable to access the first feature of the application,

wherein the first feature and each of the respective second features correspond to a respective augmented reality content item for applying to the captured image data;

downloading, by the client device and from a server, code for a first augmented reality content item corresponding to the first feature; and

activating, based on downloading the code, the first augmented reality content item with respect to the captured image data.

2. The method of claim **1**, further comprising:

displaying, in response to receiving the user input, an updated version of the custom graphic.

3. The method of claim **2**, further comprising:

providing an animation which depicts the updated version of the custom graphic as moving from the visual code to an interface element comprising the group of icons.

4. The method of claim **3**, wherein the interface element is a carousel interface for navigating the group of icons.

5. The method of claim **2**, wherein the updated version of the custom graphic corresponds to a predefined static image.

6. The method of claim **2**, wherein the updated version of the custom graphic corresponds to a predefined animation.

7. The method of claim **1**, wherein the visual code further includes a unique configuration of markings which is decodable to access to the first feature.

8. The method of claim **1**, wherein downloading the code is in response to second user input selecting the additional icon within the group of icons.

9. The method of claim **1**, wherein downloading the code is in response to receiving the user input, and

wherein activating the first augmented reality content item is in response to receiving second user input selecting the additional icon within the group of icons.

10. A device comprising:

at least one processor; and

a memory storing instructions that, when executed by the at least one processor, configure the at least one processor to perform operations comprising:

displaying, by an application running on the device, captured image data comprising a visual code, the visual code including a custom graphic and being decodable to access a first feature of the application;

receiving, by the application, user input selecting the visual code; and

in response to receiving the user input,

updating a group of icons, each icon within the group of icons being user-selectable to access a respective second feature of the application, to include an additional icon which is user-selectable to access the first feature of the application,

wherein the first feature and each of the respective second features correspond to a respective augmented reality content item for applying to the captured image data;

downloading, from a server, code for a first augmented reality content item corresponding to the first feature; and

activating, based on downloading the code, the first augmented reality content item with respect to the captured image data.

11. The device of claim **10**, the operations further comprising:

displaying, in response to receiving the user input, an updated version of the custom graphic.

12. The device of claim **11**, the operations further comprising:

providing an animation which depicts the updated version of the custom graphic as moving from the visual code to an interface element comprising the group of icons.

13. The device of claim **12**, wherein the interface element is a carousel interface for navigating the group of icons.

14. The device of claim **11**, wherein the updated version of the custom graphic corresponds to a predefined static image.

15. The device of claim **11**, wherein the updated version of the custom graphic corresponds to a predefined animation.

16. The device of claim **10**, wherein the visual code further includes a unique configuration of markings which is decodable to access to the first feature.

17. The device of claim **10**, wherein downloading the code is in response to second user input selecting the additional icon within the group of icons.

18. The device of claim **10**, wherein downloading the code is in response to receiving the user input, and

wherein activating the first augmented reality content item is in response to receiving second user input selecting the additional icon within the group of icons.

19. A non-transitory computer-readable storage medium, the computer-readable storage medium including instructions that when executed by a computer, cause the computer to perform operations comprising:

displaying, by an application running on a client device, captured image data comprising a visual code, the visual code including a custom graphic and being decodable to access a first feature of the application;

receiving, by the application, user input selecting the visual code; and

in response to receiving the user input,

updating a group of icons, each icon within the group of icons being user-selectable to access a respective sec-

ond feature of the application, to include an additional icon which is user-selectable to access the first feature of the application,

wherein the first feature and each of the respective second features correspond to a respective augmented reality content item for applying to the captured image data;

downloading, by the client device and from a server, code for a first augmented reality content item corresponding to the first feature; and

activating, based on downloading the code, the first augmented reality content item with respect to the captured image data.

20. The non-transitory computer-readable storage medium of claim **19**, the operations further comprising:

displaying, in response to receiving the user input, an updated version of the custom graphic.

* * * * *