Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2822980 A1 2012/07/05

(21) **2 822 980**

(12) **DEMANDE DE BREVET CANADIEN**
CANADIAN PATENT APPLICATION

(13) **A1**

(86) Date de dépôt PCT/PCT Filing Date: 2011/11/16
(87) Date publication PCT/PCT Publication Date: 2012/07/05
(85) Entrée phase nationale/National Entry: 2013/06/25
(86) N° demande PCT/PCT Application No.: US 2011/060935
(87) N° publication PCT/PCT Publication No.: 2012/091814
(30) Priorités/Priorities: 2010/12/30 (US61/428,759); 2010/12/30 (US61/428,717); 2010/12/30 (US61/428,734); 2011/01/05 (US61/429,863)

(51) Cl.Int./Int.Cl. *G05D 1/02* (2006.01)

(71) Demandeur/Applicant:
IROBOT CORPORATION, US

(72) Inventeurs/Inventors:
PACK, ROBERT TODD, US;
FARLOW, TIMOTHY S., US;
ROSENSTEIN, MICHAEL T., US;
HALLORAN, MICHAEL, US;
WON, CHIKYUNG, US;
SHAMLIAN, STEVEN V., US;
CHIAPPETTA, MARK, US

(74) Agent: SMART & BIGGAR

(54) Titre : SYSTEME DE ROBOT MOBILE
(54) Title: MOBILE ROBOT SYSTEM

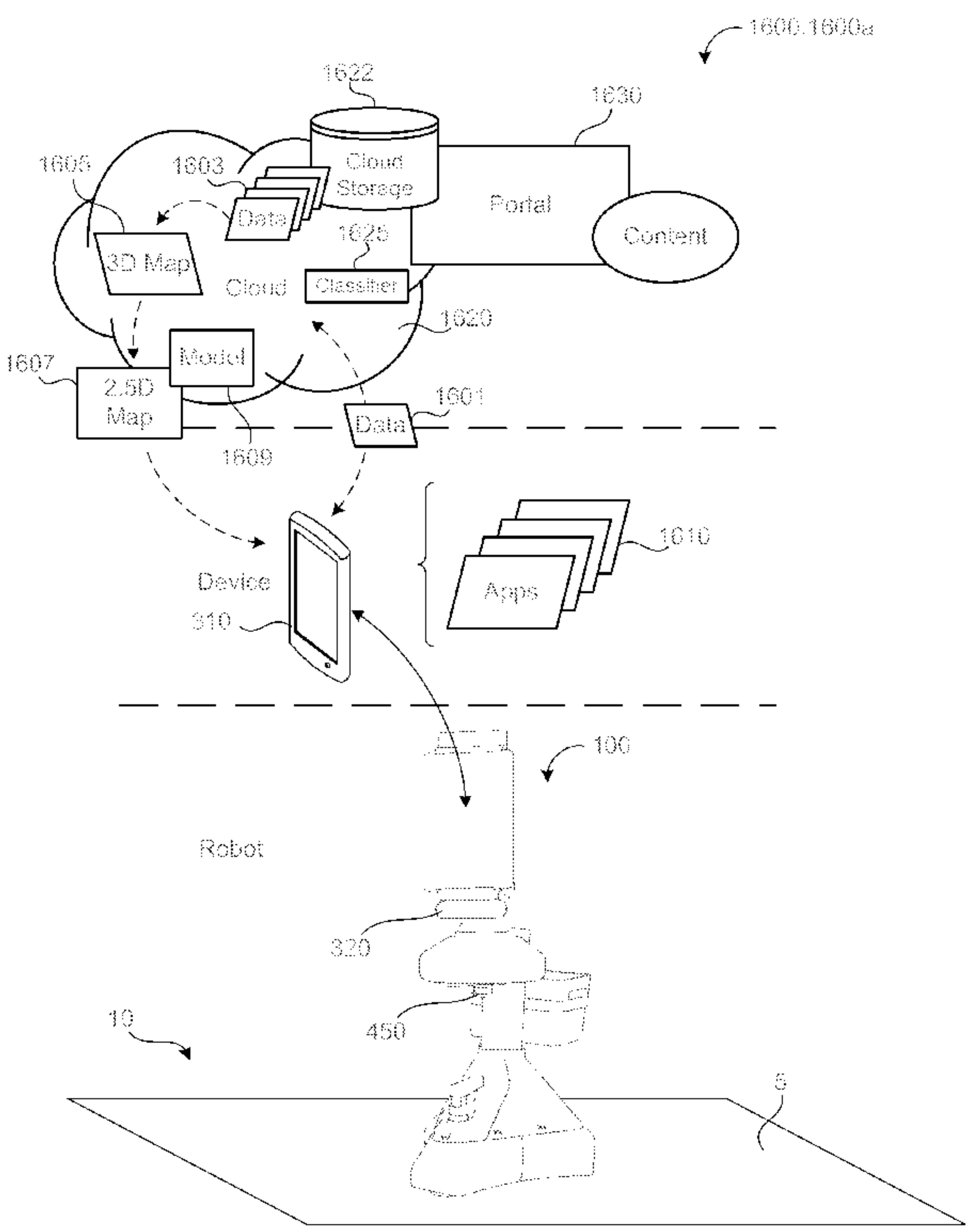

FIG. 16A

(57) Abrégé/Abstract:
A robot system (1600) includes a mobile robot (100) having a controller (500) executing a control system (510) for controlling operation of the robot, a cloud computing service (1620) in communication with the controller of the robot, and a remote

(57) Abrégé(suite)/Abstract(continued):
computing device (310) in communication with the cloud computing service. The remote computing device communicates with the robot through the cloud computing service.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
5 July 2012 (05.07.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/091814 A2

(51) International Patent Classification:
G05D 1/02 (2006.01)

(21) International Application Number:
PCT/US2011/060935

(22) International Filing Date:
16 November 2011 (16.11.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/428,734	30 December 2010 (30.12.2010)	US
61/428,717	30 December 2010 (30.12.2010)	US
61/428,759	30 December 2010 (30.12.2010)	US
61/429,863	5 January 2011 (05.01.2011)	US

(71) **Applicant** *(for all designated States except US)*: **IROBOT CORPORATION** [—/US]; 8 Crosby Drive, Bedford, MA 01730 (US).

(72) **Inventors; and**

(75) **Inventors/Applicants** *(for US only)*: **PACK, Robert, Todd** [US/US]; 27 Vieckis Drive, Nashua, NH 03062 (US). **FARLOW, Timothy, S.** [US/US]; 206 Maple St., Needham, MA 20492 (US). **ROSENSTEIN, Michael, T.** [US/US]; 25 April Lane, Westford, MA 01886 (US). **HALLORAN, Michael** [US/US]; 11 Cunningham Circle, Waltham, MA 02453 (US). **WON, Chikyung** [US/US]; 42 Windham Road, Tewksbury, MA 01876 (US). **SHAMLIAN, Steven, V.** [US/US]; 14 Hemlock Lane, Bedford, MA 01730 (US). **CHIAPPETTA, Mark** [US/US]; 7 Samuel Road, Chelmsford, MA 01824 (US).

(74) **Agents**: **KRUEGER, Brett, A.** et al.; Honigman Miller Schwartz and Cohn LLP, 350 East Michigan Avenue, Suite 300, Kalamazoo, MI 49007-3800 (US).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD,

[Continued on next page]

(54) Title: MOBILE ROBOT SYSTEM

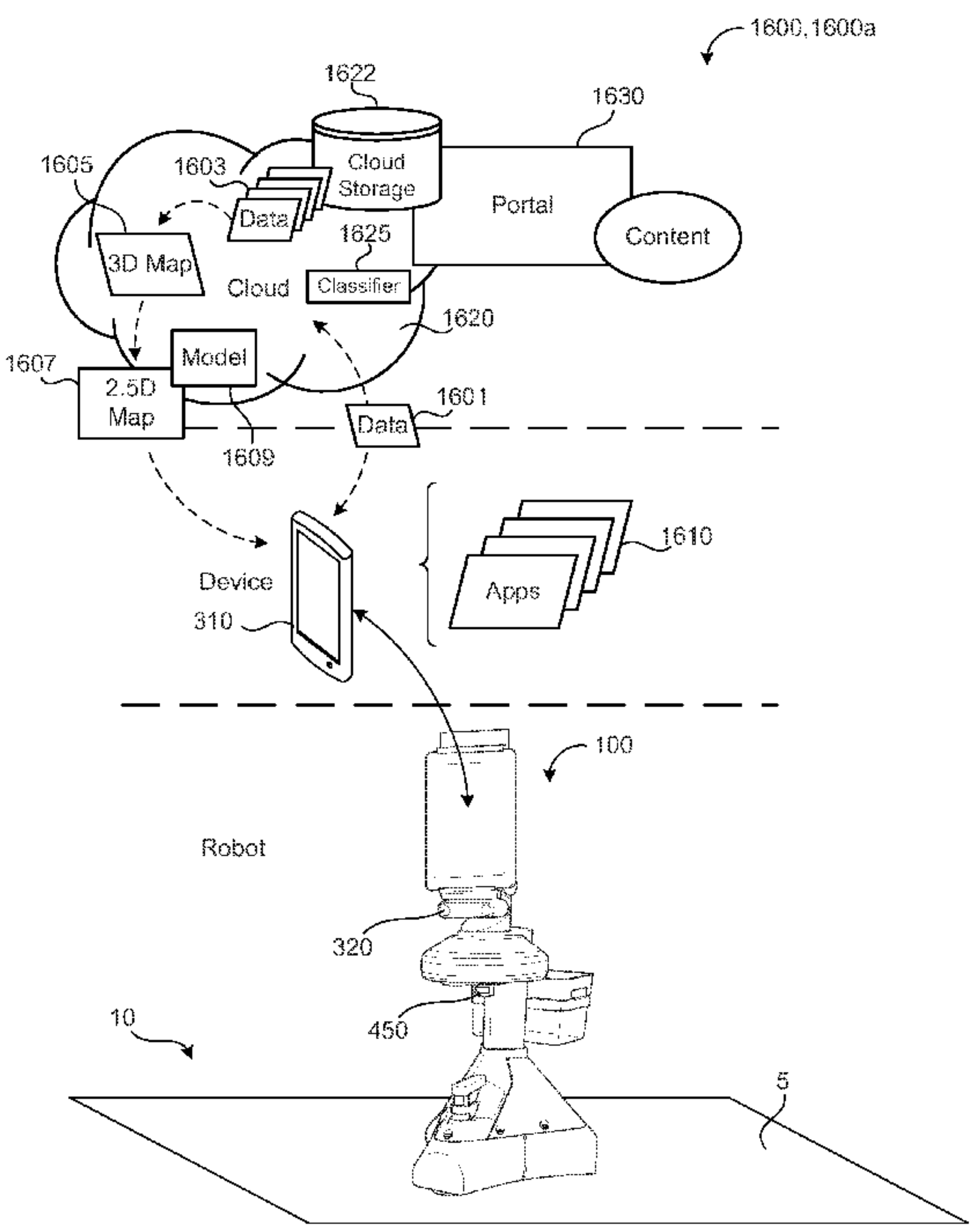

FIG. 16A

(57) Abstract: A robot system (1600) includes a mobile robot (100) having a controller (500) executing a control system (510) for controlling operation of the robot, a cloud computing service (1620) in communication with the controller of the robot, and a remote computing device (310) in communication with the cloud computing service. The remote computing device communicates with the robot through the cloud computing service.

WO 2012/091814 A2

SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

Published:

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

Mobile Robot System

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This U.S. patent application claims priority under 35 U.S.C. §119(e) to U.S. Provisional Application 61/428,717, filed on December 30, 2010; U.S. Provisional Application 61/428,734, filed on December 30, 2010; U.S. Provisional Application 61/428,759, filed on December 30, 2010; and U.S. Provisional Application 61/429,863, filed on January 5, 2011. The disclosures of these prior applications are considered part of the disclosure of this application and are hereby incorporated by reference in their entireties.

TECHNICAL FIELD

[0002] This disclosure relates to mobile robot systems incorporating cloud computing.

BACKGROUND

[0003] A robot is generally an electro-mechanical machine guided by a computer or electronic programming. Mobile robots have the capability to move around in their environment and are not fixed to one physical location. An example of a mobile robot that is in common use today is an automated guided vehicle or automatic guided vehicle (AGV). An AGV is generally a mobile robot that follows markers or wires in the floor, or uses a vision system or lasers for navigation. Mobile robots can be found in industry, military and security environments. They also appear as consumer products, for entertainment or to perform certain tasks like vacuum cleaning and home assistance.

SUMMARY

[0004] One aspect of the disclosure provides a robot system that includes a mobile robot having a controller executing a control system for controlling operation of the robot, a cloud computing service in communication with the controller of the robot, and a remote computing device in communication with the cloud computing service. The remote computing device communicates with the robot through the cloud computing service.

[0005] Implementations of the disclosure may include one or more of the following features. In some implementations, the remote computing device executes an application for producing a layout map of a robot operating environment. The remote computing device may store the layout map in external cloud storage using the cloud computing service. In some examples, the controller of the robot accesses the layout map through the cloud computing service for issuing drive commands to a drive system of the robot.

[0006] The remote computing device may execute an application (e.g., a software program or routine) providing remote teleoperation of the robot. For example, the application may provide controls for at least one of driving the robot, altering a pose of the robot, viewing video from a camera of the robot, and operating a camera of the robot (e.g., moving the camera and/or taking snapshots or pictures using the camera).

[0007] In some implementations, the remote computing device executes an application that provides video conferencing between a user of the computing device and a third party within view of a camera of the robot. The remote computing device may execute an application for scheduling usage of the robot. Moreover, the remote computing device may execute an application for monitoring usage and operation of the robot. The remote computing device may comprise a tablet computer optionally having a touch screen.

[0008] Another aspect of the disclosure provides a robot system that includes a mobile robot having a controller executing a control system for controlling operation of the robot, a computing device in communication with the controller, a cloud computing service in communication with the computing device, and a portal in communication with cloud computing service.

[0009] Implementations of the disclosure may include one or more of the following features. In some implementations, the portal comprises a web-based portal providing access to content. The portal may receive robot information from the robot through the cloud computing service. Moreover, the robot may receive user information from the portal through the cloud computing service.

[0010] In some examples, the computing device includes a touch screen (such as with a tablet computer). The computing device may execute an operating system different from an operating system of the controller. For example, the controller may execute an operating system for robot control while the computing device may execute a business enterprise operating system. In some examples, the computing device executes at least one application that collects robot information from the robot and sends the robot information to the cloud computing service.

[0011] The robot may include a base defining a vertical center axis and supporting the controller and a holonomic drive system supported by the base. The drive system has first, second, and third drive wheels, each trilaterally spaced about the vertical center axis and each having a drive direction perpendicular to a radial axis with respect to the vertical center axis. The may also include an extendable leg extending upward from the base and a torso supported by the leg. Actuation of the leg causes a change in elevation of the torso. The computing device can be detachably supported above the torso. In some examples, the robot includes a neck supported by the torso and a head supported by the neck. The neck may be capable of panning and tilting the head with respect to the torso. The head may detachably support the computing device.

[0012] Another aspect of the disclosure provides a robot system that includes a mobile robot having a controller executing a control system for controlling operation of the robot, a computing device in communication with the controller, a mediating security device controlling communications between the controller and the computing device, a cloud computing service in communication with the computing device, and a portal in communication with cloud computing service.

[0013] In some examples, the mediating security device converts communications between a computing device communication protocol of the computing device and a robot communication protocol of the robot. Moreover, the mediating security device may include an authorization chip for authorizing communication traffic between the computing device in the robot.

[0014] The computing device may communicate wirelessly with the robot controller. In some examples, the computing device is releasably attachable to the robot. An exemplary computing device includes a tablet computer.

[0015] The portal may be a web-based portal that provides access to content (e.g., news, weather, robot information, user information, etc.). In some examples, the portal receives robot information from the robot through the cloud computing service. In additional examples, the robot receives user information from the portal through the cloud computing service. The computing device may access cloud storage using the cloud computing service. The computing device may execute at least one application that collects robot information from the robot and sends the robot information to the cloud computing service.

[0016] One aspect of the disclosure provides a method of operating a mobile robot that includes receiving a layout map corresponding to an environment of the robot, moving the robot in the environment to a layout map location on the layout map, recording a robot map location on a robot map corresponding to the environment and produced by the robot, determining a distortion between the robot map and the layout map using the recorded robot map locations and the corresponding layout map locations, and applying the determined distortion to a target layout map location to determine a corresponding target robot map location.

[0017] Implementations of the disclosure may include one or more of the following features. In some implementations, the method includes receiving the layout map from a cloud computing service. The method may include producing the layout map on an application executing on a remote computing device and storing the layout map on a remote cloud storage device using the cloud computing service.

[0018] In some examples, the method includes determining a scaling size, origin mapping, and rotation between the layout map and the robot map using existing layout map locations and recorded robot map locations, and resolving a robot map location corresponding to the target layout map location. The method may further include applying an affine transformation to the determined scaling size, origin mapping, and rotation to resolve the target robot map location.

[0019] In some implementations, the method includes determining a triangulation between layout map locations that bound the target layout map location. The method may further include determining a scale, rotation, translation, and skew between a triangle mapped in the layout map and a corresponding triangle mapped in the robot map and applying the determined scale, rotation, translation, and skew to the target layout map location to determine the corresponding robot map point.

[0020] The method, in some examples, includes determining distances between all layout map locations and the target layout map location, determining a centroid of the layout map locations, determining a centroid of all recorded robot map locations, and for each layout map location, determining a rotation and a length scaling to transform a vector running from the layout map centroid to the target layout location into a vector running from the robot map centroid to the target robot map location.

[0021] The method may include producing the robot map using a sensor system of the robot. In some implementations, the method includes emitting light onto a scene of the environment, receiving reflections of the emitted light off surfaces of the scene, determining a distance of each reflecting surface, and constructing a three-dimensional depth map of the scene. The method may include emitting a speckle pattern of light onto the scene and receiving reflections of the speckle pattern from the scene. In some examples, the method includes storing reference images of the speckle pattern as reflected off a reference object in the scene, the reference images captured at different distances from the reference object. The method may further include capturing at least one target image of the speckle pattern as reflected off a target object in the scene and comparing the at least one target image with the reference images for determining a distance of the reflecting surfaces of the target object. In some examples, method includes determining a primary speckle pattern on the target object and computing at least one of a respective cross-correlation and a decorrelation between the primary speckle pattern and the speckle patterns of the reference images. The method may include maneuvering the robot with respect to the target object based on the determined distances of the reflecting surfaces of the target object.

[0022] In some implementations, the method includes determining a time-of-flight between emitting the light and receiving the reflected light and determining a distance to the reflecting surfaces of the scene. The method may include emitting the light onto the scene in intermittent pulses. Moreover, the method may include altering a frequency of the emitted light pulses.

[0023] In yet another aspect, the robot system includes a mobile robot having a controller executing a control system for controlling operation of the robot and a sensor system in communication with the controller. The robot system also includes a cloud computing service in communication with the controller of the robot. The cloud computing service receives data from the controller, processes the data, and returns processed resultant to the controller.

[0024] In some implementations, the cloud computing service at least temporarily stores the received data in cloud storage and optionally discards the stored data after processing the data. The robot, in some examples, includes a camera in communication with the controller and capable of obtaining images of a scene about the robot and/or a volumetric point cloud imaging device in communication with the controller and capable of obtaining a point cloud from a volume of space about the robot. The volume of space may include a floor plane in a direction of movement of the robot. The controller communicates image data to the cloud computing service.

[0025] The data may comprise raw sensor data and/or data having associated information from the sensor system. In some examples, the data includes image data having at least one of accelerometer data traces, odometry data, and a timestamp.

[0026] The cloud computing service may receive image data from the controller of a scene about the robot and processes the image data into a 3-D map and/or a model of the scene. Moreover, the cloud computing service may provide a 2-D height map and/or the model to the controller, where the cloud computing service computes the 2-D height map from the 3-D map. In some examples, the cloud computing service receives the image data periodically and processes the received image data after accumulating a threshold image data set.

[0027] In some implementations, the controller communicates the data to the cloud computing service wirelessly through a portable computing device (e.g., tablet computer) in communication with the controller and optionally removably attachable to the robot. The controller may buffer the data and sends the data to the cloud computing service periodically.

[0028] The sensor system may include at least one of a camera, a 3-D imaging sensor, a sonar sensor, an ultrasonic sensor, LIDAR, LADAR, an optical sensor, and an infrared sensor.

[0029] In another aspect of the disclosure, a method of operating a mobile robot includes maneuvering the robot about a scene, receiving sensor data indicative of the scene, and communicating the sensor data to a cloud computing service that processes the received sensor data and communicates a process resultant to the robot. The method further includes maneuvering the robot in the scene based on the received process resultant.

[0030] In some implementations, the method includes emitting light onto the scene about the robot and capturing images of the scene along a drive direction of the robot. The images include at least one of (a) a three-dimensional depth image, (b) an active illumination image, and (c) an ambient illumination image. The sensor data includes the images and the process resultant includes a map or a model of the scene.

[0031] The method may include emitting a speckle pattern of light onto the scene, receiving reflections of the speckle pattern from an object in the scene, and storing reference images in cloud storage of the cloud computing service of the speckle pattern as reflected off a reference object in the scene. The reference images are captured at different distances from the reference object. The method also includes capturing at least one target image of the speckle pattern as reflected off a target object in the scene and communicating the at least one target image to the cloud computing service. The cloud computing service compares the at least one target image with the reference images for determining a distance of the reflecting surfaces of the target object. In some examples, the method includes determining a primary speckle pattern on the target object and computing at least one of a respective cross-correlation and a decorrelation between the primary speckle pattern and the speckle patterns of the reference images.

[0032] The cloud computing service may at least temporarily store the received sensor data in cloud storage and optionally discard the stored sensor data after processing the data. The sensor data may include image data having associated sensor system data, which may include at least one of accelerometer data traces, odometry data, and a timestamp.

[0033] In some implementations, the cloud computing service receives image data from the robot and processes the image data into a 3-D map and/or model of the scene. The cloud computing service may provide a 2-D height map and/or the model to the robot. The cloud computing service computes the 2-D height map from the 3-D map.

[0034] The method may include periodically communicating the sensor data to the cloud computing service, which processes the received image data after accumulating a threshold sensor data set. In some examples, the method includes communicating the sensor data to the cloud computing service wirelessly through a portable computing device (e.g., tablet computer) in communication with the robot, and optionally removably attachable to the robot.

[0035] In another aspect, a method of navigating a mobile robot includes capturing a streaming sequence of dense images of a scene about the robot along a locus of motion of the robot at a real-time capture rate and associating annotations with at least some of the dense images. The method also includes sending the dense images and annotations to a remote server at a send rate, which is slower than the real-time capture rate and receiving a data set from the remote server after a processing time interval. The data set is derived from and represents at least a portion of the dense image sequence and corresponding annotations, but excludes raw image data of the sequence of dense images. The method includes moving the robot with respect to the scene based on the received data set.

[0036] The method may include sending the dense images and annotations to a local server and buffer, and then sending the dense images and annotations to the remote server at a send rate slower than the real-time capture rate. The local server and buffer may be within a relatively short-range of the robot (e.g., within between 20-100 feet or a wireless communication range).

[0037] In some implementations, the annotations include a time stamp, such as an absolute time reference corresponding to at least some of the dense images, and pose-related sensor data, which may include at least one of odometry data, accelerometer data, tilt data, and angular rate data. Annotations can be associated to the dense images that reflect hazard events captured in a time interval relative to a hazard response of the robot (e.g., avoiding a cliff, escaping from a confining situation, etc.). In additional examples, associating annotations may include associating key-frame identifiers with a subset of the dense images. The key-frame identifiers may allow identification of a dense images based on properties of the key-frame identifiers (e.g., flag, type, group, etc.).

[0038] The annotations may include a sparse set of 3-D points derived from structure and motion recovery of features tracked between dense images of the streaming sequence of dense images. The sparse set of 3-D points may be from a volumetric point imaging device on the robot. Moreover, the annotations may include camera parameters, such as a camera pose relative to individual 3-D points of the sparse set of 3-D points. Labels of traversable and non-traversable regions of the scene may be annotations for the dense images.

[0039] The data set may include one or more texture maps, such as the 2-D height map, extracted from the dense images and/or a terrain map representing features within the dense images of the scene. The data set may include a trained classifier for classifying features within new dense images captured of the scene.

[0040] In yet another aspect, a method of abstracting mobile robot environmental data includes receiving a sequence of dense images of a robot environment from a mobile robot at a receiving rate. The dense images are captured along a locus of motion of the mobile robot at a real-time capture rate. The receiving rate is slower than the real-time capture rate. The method further includes receiving annotations associated with at least some of the dense images in the sequence of dense images, and dispatching a batch processing task for reducing dense data within least some of the dense images to a data set representing at least a portion of the sequence of dense images. The method also includes transmitting the data set to the mobile robot. The data set excludes raw image data of the sequence of dense images.

[0041] In some implementations, the batch processing task includes processing the sequence of dense images into a dense 3-D model of the robot environment and processing the dense 3-D model into a terrain model for a coordinate system of 2-D location and at least one height from a floor plane. In some examples, the terrain model is for a coordinate system of 2-D location and a plurality of occupied and unoccupied height boundaries from a floor plane. For example, a terrain model if room having a table would provide data indicating upper and lower heights of an associated table top, so that the robot can determine if it can pass underneath the table.

[0042] The batch processing task may include accumulating dense image sequences corresponding to a plurality of robot environments (e.g., so that the cloud can build classifiers for identifying features of interest in any environment). As such, the batch processing task may include a plurality of classifiers and/or training one or more classifiers on the sequence of dense images. For example, the batch processing task may include associating annotations that reflect hazard events with dense images captured in a time interval relative to a hazard response of the mobile robot and training a classifier of hazard-related dense images using the associated hazard event annotations and corresponding dense images as training data, e.g., to provide a data set of model parameters for the classifier. The classifier may include at least one Support Vector Machine that constructs at least one hyperplane for classification, and the model parameters define a trained hyperplane capable of classifying a data set into hazard-related classifications. The model parameters may include sufficient parameters to define a kernel of the Support Vector Machine and a soft margin parameter

[0043] In some examples, the batch processing task includes instantiating a scalable plurality of virtual processes proportionate to a scale of the dense image sequence to be processed. At least some of the virtual processes are released after transmission of the data set to the robot. Similarly, the batch processing task may include instantiating a scalable plurality of virtual storage proportionate to a scale of the dense image sequence to be stored. At least some of virtual storage is released after transmission of the data set to the robot. Moreover, the batch processing task may include distributing a scalable plurality of virtual servers according to one of geographic proximity to the mobile robot and/or network traffic from a plurality of mobile robots.

[0044] The details of one or more implementations of the disclosure are set forth in the accompanying drawings and the description below. Other aspects, features, and advantages will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

[0045] FIG. 1 is a perspective view of an exemplary mobile human interface robot.

[0046] FIG. 2 is a schematic view of an exemplary mobile human interface robot.

[0047] FIG. 3 is an elevated perspective view of an exemplary mobile human interface robot.

[0048] FIG. 4A is a front perspective view of an exemplary base for a mobile human interface robot.

[0049] FIG. 4B is a rear perspective view of the base shown in FIG. 4A.

[0050] FIG. 4C is a top view of the base shown in FIG. 4A.

[0051] FIG. 5A is a front schematic view of an exemplary base for a mobile human interface robot.

[0052] FIG. 5B is a top schematic view of an exemplary base for a mobile human interface robot.

[0053] FIG. 5C is a front view of an exemplary holonomic wheel for a mobile human interface robot.

[0054] FIG. 5D is a side view of the wheel shown in FIG. 5C.

[0055] FIG. 6A is a front perspective view of an exemplary torso for a mobile human interface robot.

[0056] FIG. 6B is a front perspective view of an exemplary torso having touch sensing capabilities for a mobile human interface robot.

[0057] FIG. 6C is a bottom perspective view of the torso shown in FIG. 6B.

[0058] FIG. 7 is a front perspective view of an exemplary neck for a mobile human interface robot.

[0059] FIGS. 8A-8G are schematic views of exemplary circuitry for a mobile human interface robot.

[0060] FIG. 9 is a perspective view of an exemplary mobile human interface robot having detachable web pads.

[0061] FIGS. 10A-10E perspective views of people interacting with an exemplary mobile human interface robot.

[0062] FIG. 11A is a schematic view of an exemplary mobile human interface robot.

[0063] FIG. 11B is a perspective view of an exemplary mobile human interface robot having multiple sensors pointed toward the ground.

[0064] FIG. 12A is a schematic view of an exemplary imaging sensor sensing an object in a scene.

[0065] FIG. 12B is a schematic view of an exemplary arrangement of operations for operating an imaging sensor.

[0066] FIG. 12C is a schematic view of an exemplary three-dimensional (3D) speckle camera sensing an object in a scene.

[0067] FIG. 12D is a schematic view of an exemplary arrangement of operations for operating a 3D speckle camera.

[0068] FIG. 12E is a schematic view of an exemplary 3D time-of-flight (TOF) camera sensing an object in a scene.

[0069] FIG. 12F is a schematic view of an exemplary arrangement of operations for operating a 3D TOF camera.

[0070] FIG. 13 is a schematic view of an exemplary control system executed by a controller of a mobile human interface robot.

[0071] FIG. 14 is a perspective view of an exemplary mobile human interface robot receiving a human touch command.

[0072] FIG. 15 provides an exemplary telephony schematic for initiating and conducting communication with a mobile human interface robot.

[0073] FIGS. 16A-16E provide schematic views of exemplary robot system architectures.

[0074] FIG. 16F provides an exemplary arrangement of operations for a method of navigating a mobile robot.

[0075] FIG. 16G provides an exemplary arrangement of operations for a method of abstracting mobile robot environmental data.

[0076] FIG. 16H provides a schematic view of an exemplary robot system architecture.

[0077] FIG. 17A is a schematic view of an exemplary occupancy map.

[0078] FIG. 17B is a schematic view of a mobile robot having a field of view of a scene in a working area.

[0079] FIG. 18A is a schematic view of an exemplary layout map.

[0080] FIG. 18B is a schematic view of an exemplary robot map corresponding to the layout map shown in FIG. 18A.

[0081] FIG. 18C provide an exemplary arrangement of operations for operating a mobile robot to navigate about an environment using a layout map and a robot map.

[0082] FIG. 19A is a schematic view of an exemplary layout map with triangulation of type layout points.

[0083] FIG. 19B is a schematic view of an exemplary robot map corresponding to the layout map shown in FIG. 19A.

[0084] FIG. 19C provide an exemplary arrangement of operations for determining a target robot map location using a layout map and a robot map.

[0085] FIG. 20A is a schematic view of an exemplary layout map with a centroid of tight layout points.

[0086] FIG. 20B is a schematic view of an exemplary robot map corresponding to the layout map shown in FIG. 20A.

[0087] FIG. 20C provide an exemplary arrangement of operations for determining a target robot map location using a layout map and a robot map.

[0088] FIG. 21A provides an exemplary schematic view of the local perceptual space of a mobile human interface robot while stationary.

[0089] FIG. 21B provides an exemplary schematic view of the local perceptual space of a mobile human interface robot while moving.

[0090] FIG. 21C provides an exemplary schematic view of the local perceptual space of a mobile human interface robot while stationary.

[0091] FIG. 21D provides an exemplary schematic view of the local perceptual space of a mobile human interface robot while moving.

[0092] FIG. 21E provides an exemplary schematic view of a mobile human interface robot with the corresponding sensory field of view moving closely around a corner.

[0093] FIG. 21F provides an exemplary schematic view of a mobile human interface robot with the corresponding sensory field of view moving widely around a corner.

[0094] Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0095] Mobile robots can interact or interface with humans to provide a number of services that range from home assistance to commercial assistance and more. In the example of home assistance, a mobile robot can assist elderly people with everyday tasks, including, but not limited to, maintaining a medication regime, mobility assistance, communication assistance (e.g., video conferencing, telecommunications, Internet access, etc.), home or site monitoring (inside and/or outside), person monitoring, and/or providing a personal emergency response system (PERS). For commercial assistance, the mobile robot can provide videoconferencing (e.g., in a hospital setting), a point of sale terminal, interactive information/marketing terminal, etc.

[0096] Referring to FIGS. 1-2, in some implementations, a mobile robot 100 includes a robot body 110 (or chassis) that defines a forward drive direction F. The robot 100 also includes a drive system 200, an interfacing module 300, and a sensor system 400, each supported by the robot body 110 and in communication with a controller 500 that coordinates operation and movement of the robot 100. A power source 105 (e.g., battery or batteries) can be carried by the robot body 110 and in electrical communication with, and deliver power to, each of these components, as necessary. For example, the controller 500 may include a computer capable of > 1000 MIPS (million instructions per second) and the power source 1058 provides a battery sufficient to power the computer for more than three hours.

[0097] The robot body 110, in the examples shown, includes a base 120, at least one leg 130 extending upwardly from the base 120, and a torso 140 supported by the at least one leg 130. The base 120 may support at least portions of the drive system 200. The robot body 110 also includes a neck 150 supported by the torso 140. The neck 150 supports a head 160, which supports at least a portion of the interfacing module 300. The base 120 includes enough weight (e.g., by supporting the power source 105 (batteries) to maintain a low center of gravity CG_B of the base 120 and a low overall center of gravity CG_R of the robot 100 for maintaining mechanical stability.

[0098] Referring to FIGS. 3 and 4A-4C, in some implementations, the base 120 defines a trilaterally symmetric shape (e.g., a triangular shape from the top view). For example, the base 120 may include a base chassis 122 that supports a base body 124 having first, second, and third base body portions 124a, 124b, 124c corresponding to each leg of the trilaterally shaped base 120 (see e.g., FIG. 4A). Each base body portion 124a, 124b, 124c can be movably supported by the base chassis 122 so as to move independently with respect to the base chassis 122 in response to contact with an object. The trilaterally symmetric shape of the base 120 allows bump detection 360° around the robot 100. Each base body portion 124a, 124b, 124c can have an associated contact sensor e.g., capacitive sensor, read switch, etc.) that detects movement of the corresponding base body portion 124a, 124b, 124c with respect to the base chassis 122.

[0099] In some implementations, the drive system 200 provides omni-directional and/or holonomic motion control of the robot 100. As used herein the term "omni-directional" refers to the ability to move in substantially any planar direction, i.e., side-to-side (lateral), forward/back, and rotational. These directions are generally referred to herein as x, y, and θz, respectively. Furthermore, the term "holonomic" is used in a manner substantially consistent with the literature use of the term and refers to the ability to move in a planar direction with three planar degrees of freedom, i.e., two translations and one rotation. Hence, a holonomic robot has the ability to move in a planar direction at a velocity made up of substantially any proportion of the three planar velocities (forward/back, lateral, and rotational), as well as the ability to change these proportions in a substantially continuous manner.

[00100] The robot 100 can operate in human environments (e.g., environments typically designed for bipedal, walking occupants) using wheeled mobility. In some implementations, the drive system 200 includes first, second, and third drive wheels 210a, 210b, 210c equally spaced (i.e., trilaterally symmetric) about the vertical axis Z (e.g., 120 degrees apart); however, other arrangements are possible as well. Referring to FIGS. 5A and 5B, the drive wheels 210a, 210b, 210c may define a transverse arcuate rolling surface (i.e., a curved profile in a direction transverse or perpendicular to the rolling direction D_R), which may aid maneuverability of the holonomic drive system 200. Each drive wheel 210a, 210b, 210c is coupled to a respective drive motor 220a, 220b, 220c that can drive the drive wheel 210a, 210b, 210c in forward and/or reverse directions independently of the other drive motors 220a, 220b, 220c. Each drive motor 220a-c can have a respective encoder 212 (FIG. 8C), which provides wheel rotation feedback to the controller 500. In some examples, each drive wheels 210a, 210b, 210c is mounted on or near one of the three points of an equilateral triangle and having a drive direction (forward and reverse directions) that is perpendicular to an angle bisector of the respective triangle end. Driving the trilaterally symmetric holonomic base 120 with a forward driving direction F, allows the robot 100 to transition into non forward drive directions for autonomous escape from confinement or clutter and then rotating and/or translating to drive along the forward drive direction F after the escape has been resolved.

[00101] Referring to FIGS. 5C and 5D, in some implementations, each drive wheel 210 includes inboard and outboard rows 232, 234 of rollers 230, each have a rolling direction D_r perpendicular to the rolling direction D_R of the drive wheel 210. The rows 232, 234 of rollers 230 can be staggered (e.g., such that one roller 230 of the inboard row 232 is positioned equally between two adjacent rollers 230 of the outboard row 234. The rollers 230 provide infinite slip perpendicular to the drive direction the drive wheel 210. The rollers 230 define an arcuate (e.g., convex) outer surface 235 perpendicular to their rolling directions D_r, such that together the rollers 230 define the circular or substantially circular perimeter of the drive wheel 210. The profile of the rollers 230 affects the overall profile of the drive wheel 210. For example, the rollers 230 may define arcuate outer roller surfaces 235 that together define a scalloped rolling surface of the drive wheel 210 (e.g., as treads for traction). However, configuring the rollers 230 to have contours that define a circular overall rolling surface of the drive wheel 210 allows the robot 100 to travel smoothly on a flat surface instead of vibrating vertically with a wheel tread. When approaching an object at an angle, the staggered rows 232, 234 of rollers 230 (with radius r) can be used as treads to climb objects as tall or almost as tall as a wheel radius R of the drive wheel 210.

[00102] In the examples shown in FIGS. 3-5B, the first drive wheel 210a is arranged as a leading drive wheel along the forward drive direction F with the remaining two drive wheels 210b, 210c trailing behind. In this arrangement, to drive forward, the controller 500 may issue a drive command that causes the second and third drive wheels 210b, 210c to drive in a forward rolling direction at an equal rate while the first drive wheel 210a slips along the forward drive direction F. Moreover, this drive wheel arrangement allows the robot 100 to stop short (e.g., incur a rapid negative acceleration against the forward drive direction F). This is due to the natural dynamic instability of the three wheeled design. If the forward drive direction F were along an angle bisector between two forward drive wheels, stopping short would create a torque that would force the robot 100 to fall, pivoting over its two "front" wheels. Instead, travelling with one drive wheel 210a forward naturally supports or prevents the robot 100 from toppling over forward, if there is need to come to a quick stop. When accelerating from a stop, however, the controller 500 may take into account a moment of inertia I of the robot 100 from its overall center of gravity CG_R.

[00103] In some implementations of the drive system 200, each drive wheel 210a, 210b, 210 has a rolling direction D_R radially aligned with a vertical axis Z, which is orthogonal to X and Y axes of the robot 100. The first drive wheel 210a can be arranged as a leading drive wheel along the forward drive direction F with the remaining two drive wheels 210b, 210c trailing behind. In this arrangement, to drive forward, the controller 500 may issue a drive command that causes the first drive wheel 210a to drive in a forward rolling direction and the second and third drive wheels 210b, 210c to drive at an equal rate as the first drive wheel 210a, but in a reverse direction.

[00104] In other implementations, the drive system 200 can be arranged to have the first and second drive wheels 210a, 210b positioned such that an angle bisector of an angle between the two drive wheels 210a, 210b is aligned with the forward drive direction F of the robot 100. In this arrangement, to drive forward, the controller 500 may issue a drive command that causes the first and second drive wheels 210a, 210b to drive in a forward rolling direction and an equal rate, while the third drive wheel 210c drives in a reverse direction or remains idle and is dragged behind the first and second drive wheels 210a, 210b. To turn left or right while driving forward, the controller 500 may issue a command that causes the corresponding first or second drive wheel 210a, 210b to drive at relatively quicker/slower rate. Other drive system 200 arrangements can be used as well. The drive wheels 210a, 210b, 210c may define a cylindrical, circular, elliptical, or polygonal profile.

[00105] Referring again to FIGS. 1-3, the base 120 supports at least one leg 130 extending upward in the Z direction from the base 120. The leg(s) 130 may be configured to have a variable height for raising and lowering the torso 140 with respect to the base 120. In some implementations, each leg 130 includes first and second leg portions 132, 134 that move with respect to each other (e.g., telescopic, linear, and/or angular movement). Rather than having extrusions of successively smaller diameter telescopically moving in and out of each other and out of a relatively larger base extrusion, the second leg portion 134, in the examples shown, moves telescopically over the first leg portion 132, thus allowing other components to be placed along the second leg portion 134 and potentially move with the second leg portion 134 to a relatively close proximity of the base 120. The leg 130 may include an actuator assembly 136 (FIG. 8C) for moving the second leg portion 134 with respect to the first leg portion 132. The actuator assembly 136 may include a motor driver 138a in communication with a lift motor 138b and an encoder 138c, which provides position feedback to the controller 500.

[00106] Generally, telescopic arrangements include successively smaller diameter extrusions telescopically moving up and out of relatively larger extrusions at the base 120 in order to keep a center of gravity CG_L of the entire leg 130 as low as possible. Moreover, stronger and/or larger components can be placed at the bottom to deal with the greater torques that will be experienced at the base 120 when the leg 130 is fully extended. This approach, however, offers two problems. First, when the relatively smaller components are placed at the top of the leg 130, any rain, dust, or other particulate will tend to run or fall down the extrusions, infiltrating a space between the extrusions, thus obstructing nesting of the extrusions. This creates a very difficult sealing problem while still trying to maintain full mobility/articulation of the leg 130. Second, it may be desirable to mount payloads or accessories on the robot 100. One common place to mount accessories is at the top of the torso 140. If the second leg portion 134 moves telescopically in and out of the first leg portion, accessories and components could only be mounted above the entire second leg portion 134, if they need to move with the torso 140. Otherwise, any components mounted on the second leg portion 134 would limit the telescopic movement of the leg 130.

[00107] By having the second leg portion 134 move telescopically over the first leg portion 132, the second leg portion 134 provides additional payload attachment points that can move vertically with respect to the base 120. This type of arrangement causes water or airborne particulate to run down the torso 140 on the outside of every leg portion 132, 134 (e.g., extrusion) without entering a space between the leg portions 132, 134. This greatly simplifies sealing any joints of the leg 130. Moreover, payload/accessory mounting features of the torso 140 and/or second leg portion 134 are always exposed and available no matter how the leg 130 is extended.

[00108] Referring to FIGS. 3 and 6A, the leg(s) 130 support the torso 140, which may have a shoulder 142 extending over and above the base 120. In the example shown, the torso 140 has a downward facing or bottom surface 144 (e.g., toward the base) forming at least part of the shoulder 142 and an opposite upward facing or top surface 146, with a side surface 148 extending therebetween. The torso 140 may define various shapes or geometries, such as a circular or an elliptical shape having a central portion 141 supported by the leg(s) 130 and a peripheral free portion 143 that extends laterally beyond a lateral extent of the leg(s) 130, thus providing an overhanging portion that defines the downward facing surface 144. In some examples, the torso 140 defines a polygonal or other complex shape that defines a shoulder, which provides an overhanging portion that extends beyond the leg(s) 130 over the base 120.

[00109] The robot 100 may include one or more accessory ports 170 (e.g., mechanical and/or electrical interconnect points) for receiving payloads. The accessory ports 170 can be located so that received payloads do not occlude or obstruct sensors of the sensor system 400 (e.g., on the bottom surface 144 and/or the top surface 146 of the torso 140, etc.). In some implementations, as shown in FIG. 6A, the torso 140 includes one or more accessory ports 170 on a rearward portion 149 of the torso 140 for receiving a payload in the basket 360, for example, and so as not to obstruct sensors on a forward portion 147 of the torso 140 or other portions of the robot body 110.

[00110] An external surface of the torso 140 may be sensitive to contact or touching by a user, so as to receive touch commands from the user. For example, when the user touches the top surface 146 of the torso 140, the robot 100 responds by lowering a height H_T of the torso with respect to the floor (e.g., by decreasing the height H_L of the leg(s) 130 supporting the torso 140). Similarly, when the user touches the bottom surface 144 of the torso 140, the robot 100 responds by raising the torso 140 with respect to the floor (e.g., by increasing the height H_L of the leg(s) 130 supporting the torso 140). Moreover, upon receiving a user touch on forward, rearward, right or left portions of side surface 148 of the torso 140, the robot 100 responds by moving in a corresponding direction of the received touch command (e.g., rearward, forward, left, and right, respectively). The external surface(s) of the torso 140 may include a capacitive sensor in communication with the controller 500 that detects user contact.

[00111] Referring to FIGS. 6B and 6C, in some implementations, the torso 140 includes a torso body 145 having a top panel 145t, a bottom panel 145b, a front panel 145f, a back panel 145b, a right panel 145r and a left panel 145l. Each panel 145t, 145b, 145f, 145r, 145r, 145l may move independently with respect to the other panels. Moreover, each panel 145t, 145b, 145f, 145r, 145r, 145l may have an associated motion and/or contact sensor 147t, 147b, 147f, 147r, 147r, 147l in communication with the controller 500 that detects motion and/or contact with respective panel.

[00112] Referring again to FIGS. 1-3 and 7, the torso 140 supports the neck 150, which provides panning and tilting of the head 160 with respect to the torso 140. In the examples shown, the neck 150 includes a rotator 152 and a tilter 154. The rotator 152 may provide a range of angular movement θ_R (e.g., about the Z axis) of between about 90° and about 360°. Other ranges are possible as well. Moreover, in some examples, the rotator 152 includes electrical connectors or contacts that allow continuous 360° rotation of the head 150 with respect to the torso 140 in an unlimited number of rotations while maintaining electrical communication between the head 160 and the remainder of the robot 100. The tilter 154 may include the same or similar electrical connectors or contacts allow rotation of the head 160 with respect to the torso 140 while maintaining electrical communication between the head 160 and the remainder of the robot 100. The rotator 152 may include a rotator motor 152m coupled to or engaging a ring 153 (e.g., a toothed ring rack). The tilter 154 may move the head at an angle θ_T (e.g., about the Y axis) with respect to the torso 140 independently of the rotator 152. In some examples that tilter 154 includes a tilter motor 155, which moves the head 160 between an angle θ_T of ± 90° with respect to Z-axis. Other ranges are possible as well, such as ± 45°, etc. The robot 100 may be configured so that the leg(s) 130, the torso 140, the neck 150, and the head 160 stay within a perimeter of the base 120 for maintaining stable mobility of the robot 100. In the exemplary circuit schematic shown in FIG. 8F, the neck 150 includes a pan-tilt assembly 151 that includes the rotator 152 and a tilter 154 along with corresponding motor drivers 156a, 156b and encoders 158a, 158b.

[00113] The head 160 may be sensitive to contact or touching by a user, so as to receive touch commands from the user. For example, when the user pulls the head 160 forward, the head 160 tilts forward with passive resistance and then holds the position. More over, if the user pushes/pulls the head 160 vertically downward, the torso 140 may lower (via a reduction in length of the leg 130) to lower the head 160. The head 160 and/or neck 150 may include strain gauges and/or contact sensors 165 (FIG. 7) that sense user contact or manipulation.

[00114] FIGS. 8A-8G provide exemplary schematics of circuitry for the robot 100. FIGS. 8A-8C provide exemplary schematics of circuitry for the base 120, which may house the proximity sensors, such as the sonar proximity sensors 410 and the cliff proximity sensors 420, contact sensors 430, the laser scanner 440, the sonar scanner 460, and the drive system 200. The base 120 may also house the controller 500, the power source 105, and the leg actuator assembly 136. The torso 140 may house a microcontroller 140c, the microphone(s) 330, the speaker(s) 340, the scanning 3-D image sensor 450a, and a torso touch sensor system 480, which allows the controller 500 to receive and respond to user contact or touches (e.g., as by moving the torso140 with respect to the base 120, panning and/or tilting the neck 150, and/or issuing commands to the drive system 200 in response thereto). The neck 150 may house a pan-tilt assembly 151 that may include a pan motor 152 having a corresponding motor driver 156a and encoder 158a, and a tilt motor 154 having a corresponding motor driver 156b and encoder 158b. The head 160 may house one or more web pads 310 and a camera 320.

[00115] With reference to FIGS. 1-3 and 9, in some implementations, the head 160 supports one or more portions of the interfacing module 300. The head 160 may include a dock 302 for releasably receiving one or more computing tablets 310, also referred to as a web pad or a tablet PC, each of which may have a touch screen 312. The web pad 310 may be oriented forward, rearward or upward. In some implementations, web pad 310 includes a touch screen, optional I/O (e.g., buttons and/or connectors, such as micro-USB, etc.) a processor, and memory in communication with the processor. An exemplary web pad 310 includes the iPad by Apple, Inc. In some examples, the web pad and 10 functions as the controller 500 or assist the controller 500 and controlling the robot 100. In some examples, the dock 302 includes a first computing tablet 310a fixedly

attached thereto (e.g., a wired interface for data transfer at a relatively higher bandwidth, such as a gigabit rate) and a second computing tablet 310b removably connected thereto. The second web pad 310b may be received over the first web pad 310a as shown in FIG. 9, or the second web pad 310b may be received on an opposite facing side or other side of the head 160 with respect to the first web pad 310a. In additional examples, the head 160 supports a single web pad 310, which may be either fixed or removably attached thereto. The touch screen 312 may detected, monitor, and/or reproduce points of user touching thereon for receiving user inputs and providing a graphical user interface that is touch interactive. In some examples, the web pad 310 includes a touch screen caller that allows the user to find it when it has been removed from the robot 100.

[00116] In some implementations, the robot 100 includes multiple web pad docks 302 on one or more portions of the robot body 110. In the example shown in FIG. 9, the robot 100 includes a web pad dock 302 optionally disposed on the leg 130 and/or the torso 140. This allows the user to dock a web pad 310 at different heights on the robot 100, for example, to accommodate users of different height, capture video using a camera of the web pad 310 in different vantage points, and/or to receive multiple web pads 310 on the robot 100.

[00117] The interfacing module 300 may include a camera 320 disposed on the head 160 (see e.g., FIGS. 2), which can be used to capture video from elevated vantage point of the head 160 (e.g., for videoconferencing). In the example shown in FIG. 3, the camera 320 is disposed on the neck 150. In some examples, the camera 320 is operated only when the web pad 310, 310a is detached or undocked from the head 160. When the web pad 310, 310a is attached or docked on the head 160 in the dock 302 (and optionally covering the camera 320), the robot 100 may use a camera of the web pad 310a for capturing video. In such instances, the camera 320 may be disposed behind the docked web pad 310 and enters an active state when the web pad 310 is detached or undocked from the head 160 and an inactive state when the web pad 310 is attached or docked on the head 160.

[00118] The robot 100 can provide videoconferencing (e.g., at 24 fps) through the interface module 300 (e.g., using a web pad 310, the camera 320, the microphones 320, and/or the speakers 340). The videoconferencing can be multiparty. The robot 100 can provide eye contact between both parties of the videoconferencing by maneuvering the head 160 to face the user. Moreover, the robot 100 can have a gaze angle of < 5 degrees (e.g., an angle away from an axis normal to the forward face of the head 160). At least one 3-D image sensor 450 and/or the camera 320 on the robot 100 can capture life size images including body language. The controller 500 can synchronize audio and video (e.g., with the difference of <50 ms).

[00119] In the example shown in FIGS. 10A-10E, robot 100 can provide videoconferencing for people standing or sitting by adjusting the height of the web pad 310 on the head 160 and/or the camera 320 (by raising or lowering the torso 140) and/or panning and/or tilting the head 160. The camera 320 may be movable within at least one degree of freedom separately from the web pad 310. In some examples, the camera 320 has an objective lens positioned more than 3 feet from the ground, but no more than 10 percent of the web pad height from a top edge of a display area of the web pad 310. Moreover, the robot 100 can zoom the camera 320 to obtain close-up pictures or video about the robot 100. The head 160 may include one or more speakers 340 so as to have sound emanate from the head 160 near the web pad 310 displaying the videoconferencing.

[00120] In some examples, the robot 100 can receive user inputs into the web pad 310 (e.g., via a touch screen), as shown in FIG. 10E. In some implementations, the web pad 310 is a display or monitor, while in other implementations the web pad 310 is a tablet computer. The web pad 310 can have easy and intuitive controls, such as a touch screen, providing high interactivity. The web pad 310 may have a monitor display 312 (e.g., touch screen) having a display area of 150 square inches or greater movable with at least one degree of freedom.

[00121] The robot 100 can provide EMR integration, in some examples, by providing video conferencing between a doctor and patient and/or other doctors or nurses. The robot 100 may include pass-through consultation instruments. For example, the robot 100 may include a stethoscope configured to pass listening to the videoconferencing user (e.g., a doctor). In other examples, the robot includes connectors 170 that allow direct connection to Class II medical devices, such as electronic stethoscopes, otoscopes and ultrasound, to transmit medical data to a remote user (physician).

[00122] In the example shown in FIG. 10B, a user may remove the web pad 310 from the web pad dock 302 on the head 160 for remote operation of the robot 100, videoconferencing (e.g., using a camera and microphone of the web pad 310), and/or usage of software applications on the web pad 310. The robot 100 may include first and second cameras 320a, 320b on the head 160 to obtain different vantage points for videoconferencing, navigation, etc., while the web pad 310 is detached from the web pad dock 302.

[00123] Interactive applications executable on the controller 500 and/or in communication with the controller 500 may require more than one display on the robot 100. Multiple web pads 310 associated with the robot 100 can provide different combinations of "FaceTime", Telestration, HD look at this-cam (e.g., for web pads 310 having built in cameras), can act as a remote operator control unit (OCU) for controlling the robot 100 remotely, and/or provide a local user interface pad.

[00124] Referring again to FIG. 6A, the interfacing module 300 may include a microphone 330 (e.g., or micro-phone array) for receiving sound inputs and one or more speakers 340 disposed on the robot body 110 for delivering sound outputs. The microphone 330 and the speaker(s) 340 may each communicate with the controller 500. In some examples, the interfacing module 300 includes a basket 360, which may be configured to hold brochures, emergency information, household items, and other items.

[00125] Referring to FIGS. 1-4C, 11A and 11B, to achieve reliable and robust autonomous movement, the sensor system 400 may include several different types of sensors which can be used in conjunction with one another to create a perception of the robot's environment sufficient to allow the robot 100 to make intelligent decisions about actions to take in that environment. The sensor system 400 may include one or more types of sensors supported by the robot body 110, which may include obstacle detection obstacle avoidance (ODOA) sensors, communication sensors, navigation sensors, etc. For example, these sensors may include, but not limited to, proximity sensors, contact sensors, three-dimensional (3D) imaging / depth map sensors, a camera (e.g., visible light and/or infrared camera), sonar, radar, LIDAR (Light Detection And Ranging, which can entail optical remote sensing that measures properties of scattered light to find range and/or other information of a distant target), LADAR (Laser Detection and Ranging), etc.

In some implementations, the sensor system 400 includes ranging sonar sensors 410 (e.g., nine about a perimeter of the base 120), proximity cliff detectors 420, contact sensors 430, a laser scanner 440, one or more 3-D imaging/depth sensors 450, and an imaging sonar 460.

[00126] There are several challenges involved in placing sensors on a robotic platform. First, the sensors need to be placed such that they have maximum coverage of areas of interest around the robot 100. Second, the sensors may need to be placed in such a way that the robot 100 itself causes an absolute minimum of occlusion to the sensors; in essence, the sensors cannot be placed such that they are “blinded” by the robot itself. Third, the placement and mounting of the sensors should not be intrusive to the rest of the industrial design of the platform. In terms of aesthetics, it can be assumed that a robot with sensors mounted inconspicuously is more “attractive” than otherwise. In terms of utility, sensors should be mounted in a manner so as not to interfere with normal robot operation (snagging on obstacles, etc.).

[00127] In some implementations, the sensor system 400 includes a set or an array of proximity sensors 410, 420 in communication with the controller 500 and arranged in one or more zones or portions of the robot 100 (e.g., disposed on or near the base body portion 124a, 124b, 124c of the robot body 110) for detecting any nearby or intruding obstacles. The proximity sensors 410, 420 may be converging infrared (IR) emitter-sensor elements, sonar sensors, ultrasonic sensors, and/or imaging sensors (e.g., 3D depth map image sensors) that provide a signal to the controller 500 when an object is within a given range of the robot 100.

[00128] In the example shown in FIGS. 4A-4C, the robot 100 includes an array of sonar-type proximity sensors 410 disposed (e.g., substantially equidistant) around the base body 120 and arranged with an upward field of view. First, second, and third sonar proximity sensors 410a, 410b, 410c are disposed on or near the first (forward) base body portion 124a, with at least one of the sonar proximity sensors near a radially outer-most edge 125a of the first base body 124a. Fourth, fifth, and sixth sonar proximity sensors 410d, 410e, 410f are disposed on or near the second (right) base body portion 124b, with at least one of the sonar proximity sensors near a radially outer-most edge 125b of the second base body 124b. Seventh, eighth, and ninth sonar proximity sensors 410g, 410h, 410i are disposed on or near the third (right) base body portion 124c, with at least one of the sonar proximity sensors near a radially outer-most edge 125c of the third base body 124c. This configuration provides at least three zones of detection.

[00129] In some examples, the set of sonar proximity sensors 410 (e.g., 410a-410i) disposed around the base body 120 are arranged to point upward (e.g., substantially in the Z direction) and optionally angled outward away from the Z axis, thus creating a detection curtain 412 around the robot 100. Each sonar proximity sensor 410a-410i may have a shroud or emission guide 414 that guides the sonar emission upward or at least not toward the other portions of the robot body 110 (e.g., so as not to detect movement of the robot body 110 with respect to itself). The emission guide 414 may define a shell or half shell shape. In the example shown, the base body 120 extends laterally beyond the leg 130, and the sonar proximity sensors 410 (e.g., 410a-410i) are disposed on the base body 120 (e.g., substantially along a perimeter of the base body 120) around the leg 130. Moreover, the upward pointing sonar proximity sensors 410 are spaced to create a continuous or substantially continuous sonar detection curtain 412 around the leg 130. The sonar detection curtain 412 can be used to detect obstacles having elevated lateral protruding portions, such as table tops, shelves, etc.

[00130] The upward looking sonar proximity sensors 410 provide the ability to see objects that are primarily in the horizontal plane, such as table tops. These objects, due to their aspect ratio, may be missed by other sensors of the sensor system, such as the laser scanner 440 or imaging sensors 450, and as such, can pose a problem to the robot 100. The upward viewing sonar proximity sensors 410 arranged around the perimeter of the base 120 provide a means for seeing or detecting those type of objects/obstacles. Moreover, the sonar proximity sensors 410 can be placed around the widest points of the base perimeter and angled slightly outwards, so as not to be occluded or obstructed by the torso 140 or head 160 of the robot 100, thus not resulting in false positives for sensing portions of the robot 100 itself. In some implementations, the sonar proximity sensors 410 are arranged (upward and outward) to leave a volume about the torso 140 outside of a field of view of the sonar proximity sensors 410 and thus free to receive mounted payloads or accessories, such as the basket 360. The sonar proximity sensors 410 can be recessed into the base body 124 to provide visual concealment and no external features to snag on or hit obstacles.

[00131] The sensor system 400 may include or more sonar proximity sensors 410 (e.g., a rear proximity sensor 410j) directed rearward (e.g., opposite to the forward drive direction F) for detecting obstacles while backing up. The rear sonar proximity sensor 410j may include an emission guide 414 to direct its sonar detection field 412. Moreover, the rear sonar proximity sensor 410j can be used for ranging to determine a distance between the robot 100 and a detected object in the field of view of the rear sonar proximity sensor 410j (e.g., as "back-up alert"). In some examples, the rear sonar proximity sensor 410j is mounted recessed within the base body 120 so as to not provide any visual or functional irregularity in the housing form.

[00132] Referring to FIGS. 3 and 4B, in some implementations, the robot 100 includes cliff proximity sensors 420 arranged near or about the drive wheels 210a, 210b, 210c, so as to allow cliff detection before the drive wheels 210a, 210b, 210c encounter a cliff (e.g., stairs). For example, a cliff proximity sensors 420 can be located at or near each of the radially outer-most edges 125a-c of the base bodies 124a-c and in locations therebetween. In some cases, cliff sensing is implemented using infrared (IR) proximity or actual range sensing, using an infrared emitter 422 and an infrared detector 424 angled toward each other so as to have an overlapping emission and detection fields, and hence a detection zone, at a location where a floor should be expected. IR proximity sensing can have a relatively narrow field of view, may depend on surface albedo for reliability, and can have varying range accuracy from surface to surface. As a result, multiple discrete sensors can be placed about the perimeter of the robot 100 to adequately detect cliffs from multiple points on the robot 100. Moreover, IR proximity based sensors typically cannot discriminate between a cliff and a safe event, such as just after the robot 100 climbs a threshold.

[00133] The cliff proximity sensors 420 can detect when the robot 100 has encountered a falling edge of the floor, such as when it encounters a set of stairs. The controller 500 (executing a control system) may execute behaviors that cause the robot 100 to take an action, such as changing its direction of travel, when an edge is detected. In some implementations, the sensor system 400 includes one or more secondary cliff sensors (e.g., other sensors configured for cliff sensing and optionally other types of sensing). The cliff detecting proximity sensors 420 can be arranged to provide early detection of cliffs, provide data for discriminating between actual cliffs and safe events (such as climbing over thresholds), and be positioned down and out so that their field of view includes at least part of the robot body 110 and an area away from the robot body 110. In some implementations, the controller 500 executes cliff detection routine that identifies and detects an edge of the supporting work surface (e.g., floor), an increase in distance past the edge of the work surface, and/or an increase in distance between the robot body 110 and the work surface. This implementation allows: 1) early detection of potential cliffs (which may allow faster mobility speeds in unknown environments); 2) increased reliability of autonomous mobility since the controller 500 receives cliff imaging information from the cliff detecting proximity sensors 420 to know if a cliff event is truly unsafe or if it can be safely traversed (e.g., such as climbing up and over a threshold); 3) a reduction in false positives of cliffs (e.g., due to the use of edge detection versus the multiple discrete IR proximity sensors with a narrow field of view). Additional sensors arranged as "wheel drop" sensors can be used for redundancy and for detecting situations where a range-sensing camera cannot reliably detect a certain type of cliff.

[00134] Threshold and step detection allows the robot 100 to effectively plan for either traversing a climb-able threshold or avoiding a step that is too tall. This can be the same for random objects on the work surface that the robot 100 may or may not be able to safely traverse. For those obstacles or thresholds that the robot 100 determines it can climb, knowing their heights allows the robot 100 to slow down appropriately, if deemed needed, to allow for a smooth transition in order to maximize smoothness and minimize any instability due to sudden accelerations. In some implementations, threshold and step detection is based on object height above the work surface along with geometry recognition (e.g., discerning between a threshold or an electrical cable versus a blob, such as a sock). Thresholds may be recognized by edge detection. The controller 500 may receive imaging data from the cliff detecting proximity sensors 420 (or another imaging sensor on the robot 100), execute an edge detection routine, and issue a drive command based on results of the edge detection routine. The controller 500 may use pattern recognition to identify objects as well. Threshold detection allows the robot 100 to change its orientation with respect to the threshold to maximize smooth step climbing ability.

[00135] The proximity sensors 410, 420 may function alone, or as an alternative, may function in combination with one or more contact sensors 430 (e.g., bump switches) for redundancy. For example, one or more contact or bump sensors 430 on the robot body 110 can detect if the robot 100 physically encounters an obstacle. Such sensors may use a physical property such as capacitance or physical displacement within the robot 100 to determine when it has encountered an obstacle. In some implementations, each base body portion 124a, 124b, 124c of the base 120 has an associated contact sensor 430 (e.g., capacitive sensor, read switch, etc.) that detects movement of the corresponding base body portion 124a, 124b, 124c with respect to the base chassis 122 (see e.g., FIG. 4A). For example, each base body 124a-c may move radially with respect to the Z axis of the base chassis 122, so as to provide 3-way bump detection.

[00136] Referring again to FIGS. 1-4C, 11A and 11B, in some implementations, the sensor system 400 includes a laser scanner 440 mounted on a forward portion of the robot body 110 and in communication with the controller 500. In the examples shown, the laser scanner 440 is mounted on the base body 120 facing forward (e.g., having a field of view along the forward drive direction F) on or above the first base body 124a (e.g., to have maximum imaging coverage along the drive direction F of the robot). Moreover, the placement of the laser scanner on or near the front tip of the triangular base 120 means that the external angle of the robotic base (e.g., 300 degrees) is greater than a field of view 442 of the laser scanner 440 (e.g., ~285 degrees), thus preventing the base 120 from occluding or obstructing the detection field of view 442 of the laser scanner 440. The laser scanner 440 can be mounted recessed within the base body 124 as much as possible without occluding its fields of view, to minimize any portion of the laser scanner sticking out past the base body 124 (e.g., for aesthetics and to minimize snagging on obstacles).

[00137] The laser scanner 440 scans an area about the robot 100 and the controller 500, using signals received from the laser scanner 440, creates an environment map or object map of the scanned area. The controller 500 may use the object map for navigation, obstacle detection, and obstacle avoidance. Moreover, the controller 500 may use sensory inputs from other sensors of the sensor system 400 for creating object map and/or for navigation.

[00138] In some examples, the laser scanner 440 is a scanning LIDAR, which may use a laser that quickly scans an area in one dimension, as a "main" scan line, and a time-of-flight imaging element that uses a phase difference or similar technique to assign a depth to each pixel generated in the line (returning a two dimensional depth line in the plane of scanning). In order to generate a three dimensional map, the LIDAR can perform an "auxiliary" scan in a second direction (for example, by "nodding" the scanner). This mechanical scanning technique can be complemented, if not supplemented, by technologies such as the "Flash" LIDAR/LADAR and "Swiss Ranger" type focal plane imaging element sensors, techniques which use semiconductor stacks to permit time of flight calculations for a full 2-D matrix of pixels to provide a depth at each pixel, or even a series of depths at each pixel (with an encoded illuminator or illuminating laser).

[00139] The sensor system 400 may include one or more three-dimensional (3-D) image sensors 450 in communication with the controller 500. If the 3-D image sensor 450 has a limited field of view, the controller 500 or the sensor system 400 can actuate the 3-D image sensor 450a in a side-to-side scanning manner to create a relatively wider field of view to perform robust ODOA. Referring to FIGS. 1-3 and 10B, in some implementations, the robot 100 includes a scanning 3-D image sensor 450a mounted on a forward portion of the robot body 110 with a field of view along the forward drive direction F (e.g., to have maximum imaging coverage along the drive direction F of the robot). The scanning 3-D image sensor 450a can be used primarily for obstacle detection/obstacle avoidance (ODOA). In the example shown, the scanning 3-D image sensor 450a is mounted on the torso 140 underneath the shoulder 142 or on the bottom surface 144 and recessed within the torso 140 (e.g., flush or past the bottom surface 144), as shown in FIG. 3, for example, to prevent user contact with the scanning 3-D image sensor 450a. The scanning 3-D image sensor 450 can be arranged to aim substantially downward and away from the robot body 110, so as to have a downward field of view 452 in front of the robot 100 for obstacle detection and obstacle avoidance (ODOA) (e.g., with obstruction by the base 120 or other portions of the robot body 110). Placement of the scanning 3-D image sensor 450a on or near a forward edge of the torso 140 allows the field of view of the 3-D image sensor 450 (e.g., ~285 degrees) to be less than an external surface angle of the torso 140 (e.g., 300 degrees) with respect to the 3-D image sensor 450, thus preventing the torso 140 from occluding or obstructing the detection field of view 452 of the scanning 3-D image sensor 450a. Moreover, the scanning 3-D image sensor 450a (and associated actuator) can be mounted recessed within the torso 140 as much as possible without occluding its fields of view (e.g., also for aesthetics and to minimize snagging on obstacles). The distracting scanning motion of the scanning 3-D image sensor 450a is not visible to a user, creating a less distracting interaction experience. Unlike a protruding sensor or feature, the recessed scanning 3-D image sensor 450a will not tend to have unintended interactions with the environment (snagging on people, obstacles, etc.), especially when moving or scanning, as virtually no moving part extends beyond the envelope of the torso 140.

[00140] In some implementations, the sensor system 400 includes additional 3-D image sensors 450 disposed on the base body 120, the leg 130, the neck 150, and/or the head 160. In the example shown in FIG. 1, the robot 100 includes 3-D image sensors 450 on the base body 120, the torso 140, and the head 160. In the example shown in FIG. 2, the robot 100 includes 3-D image sensors 450 on the base body 120, the torso 140, and the head 160. In the example shown in FIG. 11A, the robot 100 includes 3-D image sensors 450 on the leg 130, the torso 140, and the neck 150. Other configurations are possible as well. One 3-D image sensor 450 (e.g., on the neck 150 and over the head 160) can be used for people recognition, gesture recognition, and/or videoconferencing, while another 3-D image sensor 450 (e.g., on the base 120 and/or the leg 130) can be used for navigation and/or obstacle detection and obstacle avoidance.

[00141] A forward facing 3-D image sensor 450 disposed on the neck 150 and/or the head 160 can be used for person, face, and/or gesture recognition of people about the robot 100. For example, using signal inputs from the 3-D image sensor 450 on the head 160, the controller 500 may recognize a user by creating a three-dimensional map of the viewed/captured user's face and comparing the created three-dimensional map with known 3-D images of people's faces and determining a match with one of the known 3-D facial images. Facial recognition may be used for validating users as allowable users of the robot 100. Moreover, one or more of the 3-D image sensors 450 can be used for determining gestures of person viewed by the robot 100, and optionally reacting based on the determined gesture(s) (e.g., hand pointing, waving, and or hand signals). For example, the controller 500 may issue a drive command in response to a recognized hand point in a particular direction.

[00142] The 3-D image sensors 450 may be capable of producing the following types of data: (i) a depth map, (ii) a reflectivity based intensity image, and/or (iii) a regular intensity image. The 3-D image sensors 450 may obtain such data by image pattern matching, measuring the flight time and/or phase delay shift for light emitted from a source and reflected off of a target.

[00143] In some implementations, reasoning or control software, executable on a processor (e.g., of the robot controller 500), uses a combination of algorithms executed using various data types generated by the sensor system 400. The reasoning software processes the data collected from the sensor system 400 and outputs data for making navigational decisions on where the robot 100 can move without colliding with an obstacle, for example. By accumulating imaging data over time of the robot's surroundings, the reasoning software can in turn apply effective methods to selected segments of the sensed image(s) to improve depth measurements of the 3-D image sensors 450. This may include using appropriate temporal and spatial averaging techniques.

[00144] The reliability of executing robot collision free moves may be based on: (i) a confidence level built by high level reasoning over time and (ii) a depth-perceptive sensor that accumulates three major types of data for analysis - (a) a depth image, (b) an active illumination image and (c) an ambient illumination image. Algorithms cognizant of the different types of data can be executed on each of the images obtained by the depth-perceptive imaging sensor 450. The aggregate data may improve the confidence level a compared to a system using only one of the kinds of data.

[00145] The 3-D image sensors 450 may obtain images containing depth and brightness data from a scene about the robot 100 (e.g., a sensor view portion of a room or work area) that contains one or more objects. The controller 500 may be configured to determine occupancy data for the object based on the captured reflected light from the scene. Moreover, the controller 500, in some examples, issues a drive command to the drive system 200 based at least in part on the occupancy data to circumnavigate obstacles (i.e., the object in the scene). The 3-D image sensors 450 may repeatedly capture scene depth images for real-time decision making by the controller 500 to navigate the robot 100 about the scene without colliding into any objects in the scene. For example, the speed or frequency in which the depth image data is obtained by the 3-D image sensors 450 may be controlled by a shutter speed of the 3-D image sensors 450. In addition, the controller 500 may receive an event trigger (e.g., from another sensor component of the sensor system 400, such as proximity sensor 410, 420, notifying the controller 500 of a nearby object or hazard. The controller 500, in response to the event trigger, can cause the 3-D image sensors 450 to increase a frequency at which depth images are captured and occupancy information is obtained.

[00146] Referring to FIG. 12A, in some implementations, the 3-D imaging sensor 450 includes a light source 1172 that emits light onto a scene 10, such as the area around the robot 100 (e.g., a room). The imaging sensor 450 may also include an imager 1174 (e.g., an array of light-sensitive pixels 1174p) which captures reflected light from the scene 10, including reflected light that originated from the light source 1172 (e.g., as a scene depth image). In some examples, the imaging sensor 450 includes a light source lens 1176 and/or a detector lens 1178 for manipulating (e.g., speckling or focusing) the emitted and received reflected light, respectively. The robot controller 500 or a sensor controller (not shown) in communication with the robot controller 500 receives light signals from the imager 1174 (e.g., the pixels 1174p) to determine depth information for an object 12 in the scene 10 based on image pattern matching and/or a time-of-flight characteristic of the reflected light captured by the imager 1174.

[00147] FIG. 12B provides an exemplary arrangement 1200 of operations for operating the imaging sensor 450. With additional reference to FIG. 12A, the operations include emitting 1202 light onto a scene 10 about the robot 100 and receiving 1204 reflections of the emitted light from the scene 10 on an imager (e.g., array of light-sensitive pixels). The operations further include the controller 500 receiving 1206 light detection signals from the imager, detecting 1208 one or more features of an object 12 in the scene 10 using image data derived from the light detection signals, and tracking 1210 a position of the detected feature(s) of the object 12 in the scene 10 using image depth data derived from the light detection signals. The operations may include repeating 1212 the operations of emitting 1202 light, receiving 1204 light reflections, receiving 1206 light detection signals, detecting 1208 object feature(s), and tracking 12010 a position of the object feature(s) to increase a resolution of the image data or image depth data, and/or to provide a confidence level.

[00148] The repeating 1212 operation can be performed at a relatively slow rate (e.g., slow frame rate) for relatively high resolution, an intermediate rate, or a high rate with a relatively low resolution. The frequency of the repeating 1212 operation may be adjustable by the robot controller 500. In some implementations, the controller 500 may raise or lower the frequency of the repeating 1212 operation upon receiving an event trigger. For example, a sensed item in the scene may trigger an event that causes an increased frequency of the repeating 1212 operation to sense an possibly eminent object 12 (e.g., doorway, threshold, or cliff) in the scene 10. In additional examples, a lapsed time event between detected objects 12 may cause the frequency of the repeating 1212 operation to slow down or stop for a period of time (e.g., go to sleep until awakened by another event). In some examples, the operation of detecting 1208 one or more features of an object 12 in the scene 10 triggers a feature detection event causing a relatively greater frequency of the repeating operation 1212 for increasing the rate at which image depth data is obtained. A relatively greater acquisition rate of image depth data can allow for relatively more reliable feature tracking within the scene.

[00149] The operations also include outputting 1214 navigation data for circumnavigating the object 12 in the scene 10. In some implementations, the controller 500 uses the outputted navigation data to issue drive commands to the drive system 200 to move the robot 100 in a manner that avoids a collision with the object 12.

[00150] In some implementations, the sensor system 400 detects multiple objects 12 within the scene 10 about the robot 100 and the controller 500 tracks the positions of each of the detected objects 12. The controller 500 may create an occupancy map of objects 12 in an area about the robot 100, such as the bounded area of a room. The controller 500 may use the image depth data of the sensor system 400 to match a scene 10 with a portion of the occupancy map and update the occupancy map with the location of tracked objects 12.

[00151] Referring to FIG. 12C, in some implementations, the 3-D image sensor 450 includes a three-dimensional (3D) speckle camera 1300, which allows image mapping through speckle decorrelation. The speckle camera 1300 includes a speckle emitter 1310 (e.g., of infrared, ultraviolet, and/or visible light) that emits a speckle pattern into the scene 10 (as a target region) and an imager 1320 that captures images of the speckle pattern on surfaces of an object 12 in the scene 10.

[00152] The speckle emitter 1310 may include a light source 1312, such as a laser, emitting a beam of light into a diffuser 1314 and onto a reflector 1316 for reflection, and hence projection, as a speckle pattern into the scene 10. The imager 1320 may include objective optics 1322, which focus the image onto an image sensor 1324 having an array of light detectors 1326, such as a CCD or CMOS-based image sensor. Although the optical axes of the speckle emitter 1310 and the imager 1320 are shown as being collinear, in a decorrelation mode for example, the optical axes of the speckle emitter 1310 and the imager 1320 may also be non-collinear, while in a cross-correlation mode for example, such that an imaging axis is displaced from an emission axis.

[00153] The speckle emitter 1310 emits a speckle pattern into the scene 10 and the imager 1320 captures reference images of the speckle pattern in the scene 10 at a range of different object distances Z_n from the speckle emitter 1310 (e.g., where the Z-axis can be defined by the optical axis of imager 1320). In the example shown, reference images of the projected speckle pattern are captured at a succession of planes at different, respective distances from the origin, such as at the fiducial locations marked Z_1, Z_2, Z_3, and so on. The distance between reference images, ΔZ, can be set at a threshold distance (e.g., 5 mm) or adjustable by the controller 500 (e.g., in response to triggered events). The speckle camera 1300 archives and indexes the captured reference images to the respective emission distances to allow decorrelation of the speckle pattern with distance from the speckle emitter 1310 to perform distance ranging of objects 12 captured in subsequent images. Assuming ΔZ to be roughly equal to the distance between adjacent fiducial distances Z_1, Z_2, Z_3, … , the speckle pattern on the object 12 at location Z_A can be correlated with the reference image of the speckle pattern captured at Z_2, for example.

On the other hand, the speckle pattern on the object 12 at Z_B can be correlated with the reference image at Z_3, for example. These correlation measurements give the approximate distance of the object 12 from the origin. To map the object 12 in three dimensions, the speckle camera 1300 or the controller 500 receiving information from the speckle camera 1300 can use local cross-correlation with the reference image that gave the closest match.

[00154] Other details and features on 3D image mapping using speckle ranging, via speckle cross-correlation using triangulation or decorrelation, for example, which may combinable with those described herein, can be found in PCT Patent Application PCT/IL2006/000335; the contents of which is hereby incorporated by reference in its entirety.

[00155] FIG. 12D provides an exemplary arrangement 1400 of operations for operating the speckle camera 1300. The operations include emitting 1402 a speckle pattern into the scene 10 and capturing 1404 reference images (e.g., of a reference object 12) at different distances from the speckle emitter 1310. The operations further include emitting 1406 a speckle pattern onto a target object 12 in the scene 10 and capturing 1408 target images of the speckle pattern on the object 12. The operations further include comparing 1410 the target images (of the speckled object) with different reference images to identify a reference pattern that correlates most strongly with the speckle pattern on the target object 12 and determining 1412 an estimated distance range of the target object 12 within the scene 10. This may include determining a primary speckle pattern on the object 12 and finding a reference image having speckle pattern that correlates most strongly with the primary speckle pattern on the object 12. The distance range can be determined from the corresponding distance of the reference image.

[00156] The operations optionally include constructing 1414 a 3D map of the surface of the object 12 by local cross-correlation between the speckle pattern on the object 12 and the identified reference pattern, for example, to determine a location of the object 12 in the scene. This may include determining a primary speckle pattern on the object 12 and finding respective offsets between the primary speckle pattern on multiple areas of the object 12 in the target image and the primary speckle pattern in the identified reference image so as to derive a three-dimensional (3D) map of the object. The use of solid state components for 3D mapping of a scene provides a relatively inexpensive solution for robot navigational systems.

[00157] Typically, at least some of the different, respective distances are separated axially by more than an axial length of the primary speckle pattern at the respective distances. Comparing the target image to the reference images may include computing a respective cross-correlation between the target image and each of at least some of the reference images, and selecting the reference image having the greatest respective cross-correlation with the target image.

[00158] The operations may include repeating 1416 operations 1402-1412 or operations 1406-1412, and optionally operation 1414, (e.g., continuously) to track motion of the object 12 within the scene 10. For example, the speckle camera 1300 may capture a succession of target images while the object 12 is moving for comparison with the reference images.

[00159] Other details and features on 3D image mapping using speckle ranging, which may combinable with those described herein, can be found in U.S. Patent 7,433,024; U.S. Patent Application Publication No. 2008/0106746, entitled “Depth-varying light fields for three dimensional sensing”; U.S. Patent Application Publication No. 2010/0118123, entitled “Depth Mapping Using Projected Patterns”; U.S. Patent Application Publication No. 2010/0034457, Entitled “Modeling Of Humanoid Forms From Depth Maps”; U.S. Patent Application Publication No. 2010/0020078, Entitled “Depth Mapping Using Multi-Beam Illumination”; U.S. Patent Application Publication No. 2009/0185274, Entitled “Optical Designs For Zero Order Reduction”; U.S. Patent Application Publication No. 2009/0096783, Entitled “Three-Dimensional Sensing Using Speckle Patterns”; U.S. Patent Application Publication No. 2008/0240502, Entitled “Depth Mapping Using Projected Patterns”; and U.S. Patent Application Publication No. 2008/0106746, Entitled “Depth-Varying Light Fields For Three Dimensional Sensing”; the contents of which are hereby incorporated by reference in their entireties.

[00160] Referring to FIG. 12E, in some implementations, the 3-D imaging sensor 450 includes a 3D time-of-flight (TOF) camera 1500 for obtaining depth image data. The 3D TOF camera 1500 includes a light source 1510, a complementary metal oxide semiconductor (CMOS) sensor 1520 (or charge-coupled device (CCD)), a lens 1530, and control logic or a camera controller 1540 having processing resources (and/or the robot controller 500) in communication with the light source 1510 and the CMOS sensor 1520. The light source 1510 may be a laser or light-emitting diode (LED) with an intensity that is modulated by a periodic signal of high frequency. In some examples, the light source 1510 includes a focusing lens 1512. The CMOS sensor 1520 may include an array of pixel detectors 1522, or other arrangement of pixel detectors 1522, where each pixel detector 1522 is capable of detecting the intensity and phase of photonic energy impinging upon it. In some examples, each pixel detector 1522 has dedicated detector circuitry 1524 for processing detection charge output of the associated pixel detector 1522. The lens 1530 focuses light reflected from a scene 10, containing one or more objects 12 of interest, onto the CMOS sensor 1520. The camera controller 1540 provides a sequence of operations that formats pixel data obtained by the CMOS sensor 1520 into a depth map and a brightness image. In some examples, the 3D TOF camera 1500 also includes inputs / outputs (IO) 1550 (e.g., in communication with the robot controller 500), memory 1560, and/or a clock 1570 in communication with the camera controller 1540 and/or the pixel detectors 1522 (e.g., the detector circuitry 1524).

[00161] FIG. 12F provides an exemplary arrangement 1600f of operations for operating the 3D TOF camera 1500. The operations include emitting 1602f a light pulse (e.g., infrared, ultraviolet, and/or visible light) into the scene 10 and commencing 1604f timing of the flight time of the light pulse (e.g., by counting clock pulses of the clock 1570). The operations include receiving 1606f reflections of the emitted light off one or more surfaces of an object 12 in the scene 10. The reflections may be off surfaces of the object 12 that are at different distances Z_n from the light source 1510. The reflections are received though the lens 1530 and onto pixel detectors 1522 of the CMOS sensor 1520. The operations include receiving 1608f time-of-flight for each light pulse reflection received on each corresponding pixel detector 1522 of the CMOS sensor 1520. During the roundtrip time of flight (TOF) of a light pulse, a counter of the detector circuitry 1523 of each respective pixel detector 1522 accumulates clock pulses. A larger number of accumulated clock pulses represents a longer TOF, and hence a greater distance between a light reflecting point on the imaged object 12 and the light source 1510. The operations further include determining 1610f a distance between the reflecting surface of the object

12 for each received light pulse reflection and optionally constructing 1612f a three-dimensional object surface. In some implementations, the operations include repeating 1614f operations 1602f-1610f and optionally 1612f for tracking movement of the object 12 in the scene 10.

[00162] Other details and features on 3D time-of-flight imaging, which may combinable with those described herein, can be found in U.S. Patent No. 6,323,942, entitled "CMOS Compatible 3-D Image Sensor"; U.S. Patent No. 6,515,740, entitled "Methods for CMOS-Compatible Three-Dimensional Image Sensing Using Quantum Efficiency Modulation"; and PCT Patent Application PCT/US02/16621, entitled "Method and System to Enhance Dynamic Range Conversion Usable with CMOS Three-Dimensional Imaging", the contents of which are hereby incorporated by reference in their entireties.

[00163] In some implementations, the 3-D imaging sensor 450 provides three types of information: (1) depth information (e.g., from each pixel detector 1522 of the CMOS sensor 1520 to a corresponding location on the scene 12); (2) ambient light intensity at each pixel detector location; and (3) the active illumination intensity at each pixel detector location. The depth information enables the position of the detected object 12 to be tracked over time, particularly in relation to the object's proximity to the site of robot deployment. The active illumination intensity and ambient light intensity are different types of brightness images. The active illumination intensity is captured from reflections of an active light (such as provided by the light source 1510) reflected off of the target object 12. The ambient light image is of ambient light reflected off of the target object 12. The two images together provide additional robustness, particularly when lighting conditions are poor (e.g., too dark or excessive ambient lighting).

[00164] Image segmentation and classification algorithms may be used to classify and detect the position of objects 12 in the scene 10. Information provided by these algorithms, as well as the distance measurement information obtained from the imaging sensor 450, can be used by the robot controller 500 or other processing resources. The imaging sensor 450 can operate on the principle of time-of-flight, and more specifically, on detectable phase delays in a modulated light pattern reflected from the scene 10, including techniques for modulating the sensitivity of photodiodes for filtering ambient light.

[00165] The robot 100 may use the imaging sensor 450 for 1) mapping, localization & navigation; 2) object detection & object avoidance (ODOA); 3) object hunting (e.g., to find a person); 4) gesture recognition (e.g., for companion robots); 5) people & face detection; 6) people tracking; 7) monitoring manipulation of objects by the robot 100; and other suitable applications for autonomous operation of the robot 100.

[00166] In some implementations, at least one of 3-D image sensors 450 can be a volumetric point cloud imaging device (such as a speckle or time-of-flight camera) positioned on the robot 100 at a height of greater than 1 or 2 feet above the ground and directed to be capable of obtaining a point cloud from a volume of space including a floor plane in a direction of movement of the robot (via the omni-directional drive system 200). In the examples shown in FIGS. 1 and 3, the first 3-D image sensor 450a can be positioned on the base 120 at height of greater than 1 or 2 feet above the ground (or at a height of about 1 or 2 feet above the ground) and aimed along the forward drive direction F to capture images (e.g., volumetric point cloud) of a volume including the floor while driving (e.g., for obstacle detection and obstacle avoidance). The second 3-D image sensor 450b is shown mounted on the head 160 (e.g., at a height greater than about 3 or 4 feet above the ground), so as to be capable of obtaining skeletal recognition and definition point clouds from a volume of space adjacent the robot 100. The controller 500 may execute skeletal/digital recognition software to analyze data of the captured volumetric point clouds.

[00167] Properly sensing objects 12 using the imaging sensor 450, despite ambient light conditions can be important. In many environments the lighting conditions cover a broad range from direct sunlight to bright fluorescent lighting to dim shadows, and can result in large variations in surface texture and basic reflectance of objects 12. Lighting can vary within a given location and from scene 10 to scene 10 as well. In some implementations, the imaging sensor 450 can be used for identifying and resolving people and objects 12 in all situations with relatively little impact from ambient light conditions (e.g., ambient light rejection).

[00168] In some implementations, VGA resolution of the imaging sensor 450 is 640 horizontal by 480 vertical pixels; however, other resolutions are possible as well, such. 320 x 240 (e.g., for short range sensors).

[00169] The imaging sensor 450 may include a pulse laser and camera iris to act as a bandpass filter in the time domain to look at objects 12 only within a specific range. A varying iris of the imaging sensor 450 can be used to detect objects 12 a different distances. Moreover, a pulsing higher power laser can be used for outdoor applications.

[00170] In some implementations, the robot includes a sonar scanner 460 for acoustic imaging of an area surrounding the robot 100. In the examples shown in FIGS. 1 and 3, the sonar scanner 460 is disposed on a forward portion of the base body 120.

[00171] Referring to FIGS. 1, 3B and 11B, in some implementations, the robot 100 uses the laser scanner or laser range finder 440 for redundant sensing, as well as a rear-facing sonar proximity sensor 410j for safety, both of which are oriented parallel to the ground G. The robot 100 may include first and second 3-D image sensors 450a, 450b (depth cameras) to provide robust sensing of the environment around the robot 100. The first 3-D image sensor 450a is mounted on the torso 140 and pointed downward at a fixed angle to the ground G. By angling the first 3-D image sensor 450a downward, the robot 100 receives dense sensor coverage in an area immediately forward or adjacent to the robot 100, which is relevant for short-term travel of the robot 100 in the forward direction. The rear-facing sonar 410j provides object detection when the robot travels backward. If backward travel is typical for the robot 100, the robot 100 may include a third 3D image sensor 450 facing downward and backward to provide dense sensor coverage in an area immediately rearward or adjacent to the robot 100.

[00172] The second 3-D image sensor 450b is mounted on the head 160, which can pan and tilt via the neck 150. The second 3-D image sensor 450b can be useful for remote driving since it allows a human operator to see where the robot 100 is going. The neck 150 enables the operator tilt and/or pan the second 3-D image sensor 450b to see both close and distant objects. Panning the second 3-D image sensor 450b increases an associated horizontal field of view. During fast travel, the robot 100 may tilt the second 3-D image sensor 450b downward slightly to increase a total or combined field of view of both 3-D image sensors 450a, 450b, and to give sufficient time for the robot 100 to

avoid an obstacle (since higher speeds generally mean less time to react to obstacles). At slower speeds, the robot 100 may tilt the second 3-D image sensor 450b upward or substantially parallel to the ground G to track a person that the robot 100 is meant to follow. Moreover, while driving at relatively low speeds, the robot 100 can pan the second 3-D image sensor 450b to increase its field of view around the robot 100. The first 3-D image sensor 450a can stay fixed (e.g., not moved with respect to the base 120) when the robot is driving to expand the robot's perceptual range.

[00173] In some implementations, at least one of 3-D image sensors 450 can be a volumetric point cloud imaging device (such as a speckle or time-of-flight camera) positioned on the robot 100 at a height of greater than 1 or 2 feet above the ground (or at a height of about 1 or 2 feet above the ground) and directed to be capable of obtaining a point cloud from a volume of space including a floor plane in a direction of movement of the robot (via the omni-directional drive system 200). In the examples shown in FIGS. 1 and 3, the first 3-D image sensor 450a can be positioned on the base 120 at height of greater than 1 or 2 feet above the ground and aimed along the forward drive direction F to capture images (e.g., volumetric point cloud) of a volume including the floor while driving (e.g., for obstacle detection and obstacle avoidance). The second 3-D image sensor 450b is shown mounted on the head 160 (e.g., at a height greater than about 3 or 4 feet above the ground), so as to be capable of obtaining skeletal recognition and definition point clouds from a volume of space adjacent the robot 100. The controller 500 may execute skeletal/digital recognition software to analyze data of the captured volumetric point clouds.

[00174] Referring again to FIGS. 2 and 4A-4C, the sensor system 400 may include an inertial measurement unit (IMU) 470 in communication with the controller 500 to measure and monitor a moment of inertia of the robot 100 with respect to the overall center of gravity CG_R of the robot 100.

[00175] The controller 500 may monitor any deviation in feedback from the IMU 470 from a threshold signal corresponding to normal unencumbered operation. For example, if the robot begins to pitch away from an upright position, it may be “clothes lined” or otherwise impeded, or someone may have suddenly added a heavy payload. In these instances, it may be necessary to take urgent action (including, but not limited to, evasive maneuvers, recalibration, and/or issuing an audio/visual warning) in order to assure safe operation of the robot 100.

[00176] Since robot 100 may operate in a human environment, it may interact with humans and operate in spaces designed for humans (and without regard for robot constraints). The robot 100 can limit its drive speeds and accelerations when in a congested, constrained, or highly dynamic environment, such as at a cocktail party or busy hospital. However, the robot 100 may encounter situations where it is safe to drive relatively fast, as in a long empty corridor, but yet be able to decelerate suddenly, as when something crosses the robots' motion path.

[00177] When accelerating from a stop, the controller 500 may take into account a moment of inertia of the robot 100 from its overall center of gravity CG_R to prevent robot tipping. The controller 500 may use a model of its pose, including its current moment of inertia. When payloads are supported, the controller 500 may measure a load impact on the overall center of gravity CG_R and monitor movement of the robot moment of inertia. For example, the torso 140 and/or neck 150 may include strain gauges to measure strain. If this is not possible, the controller 500 may apply a test torque command to the drive wheels 210 and measure actual linear and angular acceleration of the robot using the IMU 470, in order to experimentally determine safe limits.

[00178] During a sudden deceleration, a commanded load on the second and third drive wheels 210b, 210c (the rear wheels) is reduced, while the first drive wheel 210a (the front wheel) slips in the forward drive direction and supports the robot 100. If the loading of the second and third drive wheels 210b, 210c (the rear wheels) is asymmetrical, the robot 100 may "yaw" which will reduce dynamic stability. The IMU 470 (e.g., a gyro) can be used to detect this yaw and command the second and third drive wheels 210b, 210c to reorient the robot 100.

[00179] Referring to FIGS. 3-4C and 6A, in some implementations, the robot 100 includes multiple antennas. In the examples shown, the robot 100 includes a first antenna 490a and a second antenna 490b both disposed on the base 120 (although the antennas may be disposed at any other part of the robot 100, such as the leg 130, the torso 140, the neck 150, and/or the head 160). The use of multiple antennas provide robust signal reception and transmission. The use of multiple antennas provides the robot 100 with multiple-input and multiple-output, or MIMO, which is the use of multiple antennas for a transmitter and/or a receiver to improve communication performance. MIMO offers significant increases in data throughput and link range without additional bandwidth or transmit power. It achieves this by higher spectral efficiency (more bits per second per hertz of bandwidth) and link reliability or diversity (reduced fading). Because of these properties, MIMO is an important part of modern wireless communication standards such as IEEE 802.11n (Wifi), 4G, 3GPP Long Term Evolution, WiMAX and HSPA+. Moreover, the robot 100 can act as a Wi-Fi bridge, hub or hotspot for other electronic devices nearby. The mobility and use of MIMO of the robot 100 can allow the robot to come a relatively very reliable Wi-Fi bridge.

[00180] MIMO can be sub-divided into three main categories, pre-coding, spatial multiplexing or SM, and diversity coding. Pre-coding is a type of multi-stream beam forming and is considered to be all spatial processing that occurs at the transmitter. In (single-layer) beam forming, the same signal is emitted from each of the transmit antennas with appropriate phase (and sometimes gain) weighting such that the signal power is maximized at the receiver input. The benefits of beam forming are to increase the received signal gain, by making signals emitted from different antennas add up constructively, and to reduce the multipath fading effect. In the absence of scattering, beam forming can result in a well defined directional pattern. When the receiver has multiple antennas, the transmit beam forming cannot simultaneously maximize the signal level at all of the receive antennas, and pre-coding with multiple streams can be used. Pre-coding may require knowledge of channel state information (CSI) at the transmitter.

[00181] Spatial multiplexing requires a MIMO antenna configuration. In spatial multiplexing, a high rate signal is split into multiple lower rate streams and each stream is transmitted from a different transmit antenna in the same frequency channel. If these signals arrive at the receiver antenna array with sufficiently different spatial signatures, the receiver can separate these streams into (almost) parallel channels. Spatial multiplexing is a very powerful technique for increasing channel capacity at higher signal-to-noise ratios (SNR). The maximum number of spatial streams is limited by the lesser in the number of antennas at the transmitter or receiver. Spatial multiplexing can be used with or without transmit channel knowledge. Spatial multiplexing can also be used for simultaneous transmission to multiple receivers, known as space-division multiple access. By scheduling receivers with different spatial signatures, good separability can be assured.

[00182] Diversity Coding techniques can be used when there is no channel knowledge at the transmitter. In diversity methods, a single stream (unlike multiple streams in spatial multiplexing) is transmitted, but the signal is coded using techniques called space-time coding. The signal is emitted from each of the transmit antennas with full or near orthogonal coding. Diversity coding exploits the independent fading in the multiple antenna links to enhance signal diversity. Because there is no channel knowledge, there is no beam forming or array gain from diversity coding. Spatial multiplexing can also be combined with pre-coding when the channel is known at the transmitter or combined with diversity coding when decoding reliability is in trade-off.

[00183] In some implementations, the robot 100 includes a third antenna 490c and/or a fourth antenna 490d and the torso 140 and/or the head 160, respectively (see e.g., FIG. 3). In such instances, the controller 500 can determine an antenna arrangement (e.g., by moving the antennas 490a-d, as by raising or lowering the torso 140 and/or rotating and/or tilting the head 160) that achieves a threshold signal level for robust communication. For example, the controller 500 can issue a command to elevate the third and fourth antennas 490c, 490d by raising a height of the torso 140. Moreover, the controller 500 can issue a command to rotate and/or the head 160 to further orient the fourth antenna 490d with respect to the other antennas 490a-c.

[00184] Referring to FIG. 13, in some implementations, the controller 500 executes a control system 510, which includes a control arbitration system 510a and a behavior system 510b in communication with each other. The control arbitration system 510a allows applications 520 to be dynamically added and removed from the control system 510, and facilitates allowing applications 520 to each control the robot 100 without needing to know about any other applications 520. In other words, the control arbitration system 510a provides a simple prioritized control mechanism between applications 520 and resources 530 of the robot 100. The resources 530 may include the drive system 200, the sensor system 400, and/or any payloads or controllable devices in communication with the controller 500.

[00185] The applications 520 can be stored in memory of or communicated to the robot 100, to run concurrently on (e.g., a processor) and simultaneously control the robot 100. The applications 520 may access behaviors 600 of the behavior system 510b. The independently deployed applications 520 are combined dynamically at runtime and to share robot resources 530 (e.g., drive system 200, arm(s), head(s), etc.) of the robot 100. A low-level policy is implemented for dynamically sharing the robot resources 530 among the applications 520 at run-time. The policy determines which application 520 has control of the robot resources 530 required by that application 520 (e.g. a priority hierarchy among the applications 520). Applications 520 can start and stop dynamically and run completely independently of each other. The control system 510 also allows for complex behaviors 600 which can be combined together to assist each other.

[00186] The control arbitration system 510a includes one or more resource controllers 540, a robot manager 550, and one or more control arbiters 560. These components do not need to be in a common process or computer, and do not need to be started in any particular order. The resource controller 540 component provides an interface to the control arbitration system 510a for applications 520. There is an instance of this component for every application 520. The resource controller 540 abstracts and encapsulates away the complexities of authentication, distributed resource control arbiters, command buffering, and the like. The robot manager 550 coordinates the prioritization of applications 520, by controlling which application 520 has exclusive control of any of the robot resources 530 at any particular time. Since this is the central coordinator of information, there is only one instance of the robot manager 550 per robot. The robot manager 550 implements a priority policy, which has a linear prioritized order of the resource controllers 540, and keeps track of the resource control arbiters 560 that provide hardware control. The control arbiter 560 receives the commands from every application 520 and generates a single command based on the applications' priorities and publishes it for its associated resources 530. The control arbiter 560 also receives state feedback from its associated resources 530 and sends it back up to the applications 520. The robot resources 530 may be a network of functional modules (e.g. actuators, drive systems, and groups thereof) with one or more hardware controllers. The commands of the control arbiter 560 are specific to the resource 530 to carry out specific actions.

[00187] A dynamics model 570 executable on the controller 500 can be configured to compute the center for gravity (CG), moments of inertia, and cross products of inertia of various portions of the robot 100 for the assessing a current robot state. The dynamics model 570 may also model the shapes, weight, and/or moments of inertia of these components. In some examples, the dynamics model 570 communicates with the inertial moment unit 470 (IMU) or portions of one (e.g., accelerometers and/or gyros) disposed on the robot 100 and in communication with the controller 500 for calculating the various center of gravities of the robot 100. The dynamics model 570 can be used by the controller 500, along with other programs 520 or behaviors 600 to determine operating envelopes of the robot 100 and its components.

[00188] Each application 520 has an action selection engine 580 and a resource controller 540, one or more behaviors 600 connected to the action selection engine 580, and one or more action models 590 connected to action selection engine 580. The behavior system 510b provides predictive modeling and allows the behaviors 600 to collaboratively decide on the robot's actions by evaluating possible outcomes of robot actions. In some examples, a behavior 600 is a plug-in component that provides a hierarchical, state-full evaluation function that couples sensory feedback from multiple sources with a-priori limits and information into evaluation feedback on the allowable actions of the robot. Since the behaviors 600 are pluggable into the application 520 (e.g., residing inside or outside of the application 520), they can be removed and added without having to modify the application 520 or any other part of the control system 510. Each behavior 600 is a standalone policy. To make behaviors 600 more powerful, it is possible to attach the output of multiple behaviors 600 together into the input of another so that you can have complex combination functions. The behaviors 600 are intended to implement manageable portions of the total cognizance of the robot 100.

[00189] The action selection engine 580 is the coordinating element of the control system 510 and runs a fast, optimized action selection cycle (prediction/correction cycle) searching for the best action given the inputs of all the behaviors 600. The action selection engine 580 has three phases: nomination, action selection search, and completion. In the nomination phase, each behavior 600 is notified that the action selection cycle has started and is provided with the cycle start time, the current state, and

limits of the robot actuator space. Based on internal policy or external input, each behavior 600 decides whether or not it wants to participate in this action selection cycle. During this phase, a list of active behavior primitives is generated whose input will affect the selection of the commands to be executed on the robot 100.

[00190] In the action selection search phase, the action selection engine 580 generates feasible outcomes from the space of available actions, also referred to as the action space. The action selection engine 580 uses the action models 590 to provide a pool of feasible commands (within limits) and corresponding outcomes as a result of simulating the action of each command at different time steps with a time horizon in the future. The action selection engine 580 calculates a preferred outcome, based on the outcome evaluations of the behaviors 600, and sends the corresponding command to the control arbitration system 510a and notifies the action model 590 of the chosen command as feedback.

[00191] In the completion phase, the commands that correspond to a collaborative best scored outcome are combined together as an overall command, which is presented to the resource controller 540 for execution on the robot resources 530. The best outcome is provided as feedback to the active behaviors 600, to be used in future evaluation cycles.

[00192] Received sensor signals from the sensor system 400 can cause interactions with one or more behaviors 600 to execute actions. For example, using the control system 510, the controller 500 selects an action (or move command) for each robotic component (e.g., motor or actuator) from a corresponding action space (e.g., a collection of possible actions or moves for that particular component) to effectuate a coordinated move of each robotic component in an efficient manner that avoids collisions with itself and any objects about the robot 100, which the robot 100 is aware of. The controller 500 can issue a coordinated command over robot network, such as an EtherIO network, as described in U.S. Serial No. 61/305,069, filed February 16, 2010, the entire contents of which are hereby incorporated by reference.

[00193] The control system 510 may provide adaptive speed/acceleration of the drive system 200 (e.g., via one or more behaviors 600) in order to maximize stability of the robot 100 in different configurations/positions as the robot 100 maneuvers about an area.

[00194] In some implementations, the controller 500 issues commands to the drive system 200 that propels the robot 100 according to a heading setting and a speed setting. One or behaviors 600 may use signals received from the sensor system 400 to evaluate predicted outcomes of feasible commands, one of which may be elected for execution (alone or in combination with other commands as an overall robot command) to deal with obstacles. For example, signals from the proximity sensors 410 may cause the control system 510 to change the commanded speed or heading of the robot 100. For instance, a signal from a proximity sensor 410 due to a nearby wall may result in the control system 510 issuing a command to slow down. In another instance, a collision signal from the contact sensor(s) due to an encounter with a chair may cause the control system 510 to issue a command to change heading. In other instances, the speed setting of the robot 100 may not be reduced in response to the contact sensor; and/or the heading setting of the robot 100 may not be altered in response to the proximity sensor 410.

[00195] The behavior system 510b may include a speed behavior 600 (e.g., a behavioral routine executable on a processor) configured to adjust the speed setting of the robot 100 and a heading behavior 600 configured to alter the heading setting of the robot 100. The speed and heading behaviors 600 may be configured to execute concurrently and mutually independently. For example, the speed behavior 600 may be configured to poll one of the sensors (e.g., the set(s) of proximity sensors 410, 420), and the heading behavior 600 may be configured to poll another sensor (e.g., the kinetic bump sensor).

[00196] Referring to FIGS. 13 and 14, the behavior system 510b may include a torso touch teleoperation behavior 600a (e.g., a behavioral routine executable on a processor) configured to react to a user 15 touching the torso 140 for teleoperation (e.g., guiding the robot 100). The torso touch teleoperation behavior 600a may become active when the sensor system 400 detects that the torso has received contact (e.g., human contact) for at least a threshold time period (e.g., 0.25 seconds). For example, the motion and/or contact sensor 147t, 147b, 147f, 147r, 147r, 147l in communication with the controller 500 and associated with the corresponding top panel 145t, a bottom panel 145b, a front panel 145f, a back panel 145b, a right panel 145r and a left panel 145l of the torso body 145 can detect motion and/or contact with the respective panel, as shown in FIGS. 6B and 6C. Once active, the torso touch teleoperation behavior 600a receives a contact force direction (e.g., as sensed and computed from an ellipse location of the touch) and issues a velocity command to the drive system 200 in local X/Y coordinates (taking advantage of the holonomic mobility). Obstacle detection and obstacle avoidance behaviors may be turned off for while the torso touch teleoperation behavior 600a is active. If the sensed touch location, force, or direction changes, the torso touch teleoperation behavior 600a changes the velocity command to correspond with the sensed contact force direction. The torso touch teleoperation behavior 600a may execute a stop routine when the sensor system 400 no longer senses contact with the robot 100 for a threshold period of time (e.g., 2 seconds). The stop routine may cause the drive system 200 to stop driving after about 0.5 seconds if the sensor system 400 no longer senses contact with the robot 100 (e.g., with the torso 140). The torso touch teleoperation behavior 600a may provide a delay in stopping the robot 100 to allow moving the touch point without having to wait for a trigger period of time.

[00197] The torso touch teleoperation behavior 600a may issue assisted drive commands to the drive system 200 that allow the user to push the robot 100 while receiving drive assistance from the drive system 200 (e.g., partial velocity commands that by themselves cannot move the robot 100, but assist movement of the robot 100 by the user).

[00198] The torso touch teleoperation behavior 600a may receive sensor signals from the touch sensor system 480 (e.g., buttons, capacitive sensors, contact sensors, etc.), a portion of which may be disposed on the torso 140 (and elsewhere on the robot 100, such as the head 160). The torso touch teleoperation behavior 600a may position the torso 140 at a height H_T of between 3 and 5 feet from the ground G, so as to place at least a portion of the touch sensor system 480 at an accessible height for a typical user.

[00199] In some implementations, the torso touch teleoperation behavior 600a recognizes user touching to place the robot 100 and particular pose. For example, when the user 15 pushes down on the torso 140, the sensor system 400 detects the downward force on the torso 140 and sends corresponding signals to the controller 500. The torso touch teleoperation behavior 600a receives indication of the downward force on the torso 140 and causes the control system 510 to issue a command to decrease the length H_L of the leg 130, thereby lowering the height H_T of the torso 140. Similarly, when the user 15 pushes/pulls up on the torso 140, the torso touch teleoperation behavior 600a receives indication of the upward force on the torso 140 from the sensor system 400 and causes the control system 510 to issue a command to increase the length H_L of the leg 130, thereby increasing the height H_T of the torso 140.

[00200] When the user 15 pushes, pulls and/or rotates the head 160, the torso touch teleoperation behavior 600a may receive indication from the sensor system 400 (e.g., from strain gages/motion/contact sensors 165 on the neck 150) of the user action and may respond by causing the control system 510 to issue a command to move the head 160 accordingly and thereafter hold the pose.

[00201] In some implementations, the robot 100 provides passive resistance and/or active assistance to user manipulation of the robot 100. For example, the motors 138b, 152, 154 actuating the leg 130 and the neck 150 passive resistance and/or active assistance to user manipulation of the robot 100 to provide feedback to the user of the manipulation as well as assistance for moving relatively heavy components such as raising the torso 140. This allows the user to move various robotic components without having to bear the entire weight of the corresponding components.

[00202] The behavior system 510b may include a tap-attention behavior 600b (e.g., a behavioral routine executable on a processor) configured to focus attention of the robot 100 toward a user. The tap-attention behavior 600b may become active when the sensor system 400 detects that the torso 140 (or some other portion of the robot 100) has received contact (e.g., human contact) for less than a threshold time period (e.g., 0.25 seconds). Moreover, the tap-attention behavior 600b may only become active when the torso touch teleoperation behavior 600a is inactive. For example, a sensed touch on the torso 140 for 0.2 seconds will not trigger the torso touch teleoperation behavior 600a, but will trigger the tap-attention behavior 600b. The tap-attention behavior 600b may use a contact location on the torso 140 and cause the head 160 to tilt and/or pan (via actuation of the neck 150) to look at the user. A stop criteria for the behavior 600b can be reached when the head 160 reaches a position where it is looking in the direction of the touch location.

[00203] In some implementations, the behavior system 510b includes a tap-stop behavior 600c (e.g., a behavioral routine executable on a processor) configured to stop the drive system 200 from driving (e.g., bring the robot 100 to a stop). The tap-stop behavior 600c may become active when the sensor system 400 detects that the torso 140 has received contact (e.g., human contact) and issues a zero velocity drive command to the drive system 200, cancelling any previous drive commands. If the robot is driving and the user wants it to stop, the user can tap the torso 140 (or some other portion of the robot 100) or a touch sensor. In some examples, the tap-stop behavior 600c can only be activated if higher priority behaviors, such as the torso touch teleoperation behavior 600a and the tap-attention behavior 600b, are not active. The tap-stop behavior 600c may end with the sensor system 400 no longer detects touching on the torso 140 (or elsewhere on the robot 100).

[00204] In some implementations, the robot 100 includes a mediating security device 350 (FIG. 9), also referred to as a bridge, for allowing communication between a web pad 310 and the controller 500 (and/or other components of the robot 100). For example, the bridge 350 may convert communications of the web pad 310 from a web pad communication protocol to a robot communication protocol (e.g., Ethernet having a gigabit capacity). The bridge 350 may authenticate the web pad 310 and provided communication conversion between the web pad 310 and the controller 500. In some examples, the bridge 350 includes an authorization chip 352 which authorizes/validates any communication traffic between the web pad 310 and the robot 100. The bridge 350 may notify the controller 500 when it has checked an authorized a web pad 310 trying to communicate with the robot 100. Moreover, after authorization, the bridge 350 notify the web pad 310 of the communication authorization. The bridge 350 may be disposed on the neck 150 or head (as shown in FIGS. 2 and 3) or elsewhere on the robot 100.

[00205] The Session Initiation Protocol (SIP) is an IETF-defined signaling protocol, widely used for controlling multimedia communication sessions such as voice and video calls over Internet Protocol (IP). The protocol can be used for creating, modifying and terminating two-party (unicast) or multiparty (multicast) sessions including one or several media streams. The modification can involve changing addresses or ports, inviting more participants, and adding or deleting media streams. Other feasible application examples include video conferencing, streaming multimedia distribution, instant messaging, presence information, file transfer, etc. Voice over Internet Protocol (Voice over IP, VoIP) is part of a family of methodologies, communication protocols, and transmission technologies for delivery of voice communications and multimedia sessions over Internet Protocol (IP) networks, such as the Internet. Other terms frequently encountered and often used synonymously with VoIP are IP telephony, Internet telephony, voice over broadband (VoBB), broadband telephony, and broadband phone.

[00206] FIG. 15 provides a telephony example that includes interaction with the bridge 350 for initiating and conducting communication through the robot 100. An SIP of Phone A places a call with the SIP application server. The SIP invokes a dial function of the VoIP, which causes a HTTP post request to be sent to a VoIP web server. The HTTP Post request may behave like a callback function. The SIP application server sends a ringing to phone A, indicating that the call has been initiated. A VoIP server initiates a call via a PSTN to a callback number contained in the HTTP post request. The callback number terminates on a SIP DID provider which is configured to route calls back to the SIP application server. The SIP application server matches an incoming call with the original call of phone A and answers both calls with an OK response. A media session is established between phone A and the SIP DID provider. Phone A may hear an artificial ring generated by the VoIP. Once the VoIP has verified that the callback leg has been answered, it initiates the PSTN call to the destination, such as the robot 100 (via the bridge 350). The robot 100 answers the call and the VoIP server bridges the media from the SIP DID provider with the media from the robot 100.

[00207] FIGS. 16A-16D provide schematic views of exemplary robot system architectures 1600, 1600a-d, which may include the robot 100 (or a portion thereof, such as the controller 500 or drive system 200), a computing device 310 (detachable or fixedly attached to the head 160), a cloud 1620 (for cloud computing), and a portal 1630.

[00208] The robot 100 can provide various core robot features, which may include: mobility (e.g., the drive system 200); a reliable, safe, secure robot intelligence system, such as a control system 510 (FIG. 13) executed on the controller 500, the power source 105, the sensing system 400, and optional manipulation with a manipulator in communication with the controller 500. The control system 510 can provide heading and speed control, body pose control, navigation, and core robot applications. The sensing system 400 can provide vision (e.g., via a camera 320), depth map imaging (e.g., via a 3-

D imaging sensor 450), collision detection, obstacle detection and obstacle avoidance, and/or inertial measurement (e.g., via an inertial measurement unit 470).

[00209] The computing device 310 may be a tablet computer, portable electronic device, such as phone or personal digital assistant, or a dumb tablet or display (e.g., a tablet that acts as a monitor for an atom-scale PC in the robot body 110). In some examples, the tablet computer can have a touch screen for displaying a user interface and receiving user inputs. The computing device 310 may execute one or more robot applications 1610, which may include software applications (e.g., stored in memory and executable on a processor) for security, medicine compliance, telepresence, behavioral coaching, social networking, active alarm, home management, etc. The computing device 310 may provide communication capabilities (e.g., secure wireless connectivity and/or cellular communication), refined application development tools, speech recognition, and person or object recognition capabilities. The computing device 310, in some examples utilizes an interaction/COMS featured operating system, such as Android provided by Google, Inc., iPad OS provided by Apple, Inc., other smart phone operating systems, or specialized robot operating systems, such as RSS A2.

[00210] The cloud 1620 provides cloud computing and/or cloud storage capabilities. Cloud computing may provide Internet-based computing, whereby shared servers provide resources, software, and data to computers and other devices on demand. For example, the cloud 1620 may be a cloud computing service that includes at least one server computing device, which may include a service abstraction layer and a hypertext transfer protocol wrapper over a server virtual machine instantiated thereon. The server computing device may be configured to parse HTTP requests and send HTTP responses. Cloud computing may be a technology that uses the Internet and central remote servers to maintain data and applications. Cloud computing can allow users to access and use applications 1610 without installation and access personal files at any computer with internet access. Cloud computing allows for relatively more efficient computing by centralizing storage, memory, processing and bandwidth. The cloud 1620 can provide scalable, on-demand computing power, storage, and bandwidth, while reducing robot hardware requirements (e.g., by freeing up CPU and memory usage). Robot connectivity to the cloud 1620 allows automatic data gathering of robot operation and usage histories without requiring the robot 100 to return to a base station. Moreover, continuous data collection over time can yields a wealth of data that can be mined for marketing, product development, and support.

[00211] Cloud storage 1622 can be a model of networked computer data storage where data is stored on multiple virtual servers, generally hosted by third parties. By providing communication between the robot 100 and the cloud 1620, information gathered by the robot 100 can be securely viewed by authorized users via a web based information portal.

[00212] The portal 1630 may be a web-based user portal for gathering and/or providing information, such as personal information, home status information, and robot status information. Information can be integrated with third-party information to provide additional functionality and resources to the user and/or the robot 100. The robot system architecture 1600 can facilitate proactive data collection. For example, applications 1610 executed on the computing device 310 may collect data and report on actions performed by the robot 100 and/or a person or environment viewed by the robot 100 (using the sensing system 400). This data can be a unique property of the robot 100.

[00213] In some examples, the portal 1630 is a personal portal web site on the World Wide Web. The portal 1630 may provide personalized capabilities and a pathway to other content. The portal 1630 may use distributed applications, different numbers and types of middleware and hardware, to provide services from a number of different sources. In addition, business portals 1630 may share collaboration in workplaces and provide content usable on multiple platforms such as personal computers, personal digital assistants (PDAs), and cell phones/mobile phones. Information, news, and updates are examples of content that may be delivered through the portal 1630. Personal portals 1630 can be related to any specific topic such as providing friend information on a social network or providing links to outside content that may help others.

[00214] 'Dense data' vs. 'sparse data' and 'dense features' vs. 'sparse features' are referred to herein with respect to spatial data sets. Without limiting or narrowing the meaning from that of those skill in the art would interpret such terms to mean, 'dense' vs. 'sparse' generally means many data points per spatial representation vs. few data points, and specifically may mean:

[00215] (i) in the context of 2-D image data or 3-D 'images' including 2-D data and range, 'dense' image data includes image data substantially fully populated with pixels, or capable of being rasterized to pixels with substantially no losses and/or artifacting from the original image capture (including substantially uncompressed, raw, or losslessly compressed images), while a 'sparse' image is one where the image is quantized, sampled, lossy compressed, vectorized, segmented (e.g., into superpixels, nodes, edges, surfaces, interest points, voxels), or otherwise materially reduced in fidelity from the original capture, or must be interpolated in being rasterized to pixels to re-represent an image;

[00216] (ii) in the context of 2-D or 3-D features, 'dense features' may be features that are populated in a substantially unconstrained manner, to the resolution of the detection approach - all that can be detected and recorded, and/or features that are recognized by detectors recognized to collect many features (HOG, wavelets) over a sub-image; 'sparse features' may be purposefully constrained in number, in the number of feature inputs, lateral inhibition, and/or feature selection, and/or may be recognized by detectors recognized to identify a limited number of isolated points in an image (Harris corner, edges, Shi-Tomasi).

[00217] With respect to 3-D environment structure, the robot 100 may acquire images, such as dense images 1611, of a scene 10 about the robot 100 while maneuvering about a work surface 5. In some implementations, the robot 100 uses a camera 320 and/or an imaging sensor 450 (e.g., volumetric point cloud imaging device) for obtaining the dense images 1611. The controller 500, which is in communication with the camera 320 and/or the imaging sensor 450 may associate information 1613 with the dense images 1611 (e.g., annotate or tag the dense images 1611 with data), such as accelerometer data traces, odometry data, and/or other data from the sensor system 400 along with timestamps. In some examples, the robot 100 captures a streaming sequence 1615 of dense images 1615 and annotates the dense image sequence 1615 with annotation data 1613, providing an annotated dense image sequence 1615a. The robot 100 may transmit image data 1601 (e.g., via the controller 500 or web pad 310) periodically to the cloud storage 1622, which can accumulate a potentially very large image data set 1603 over a work time. The image data 1601 may be raw sensor data (e.g., a point cloud or signal or the dense image sequence 1615) or tagged data, such as the annotated dense image sequence 1615a (e.g., a

data object having properties or attributes, such as with Java Script Object Notation (JSON) objects). The cloud service 1620 may process the received image data 1601 (e.g., dense image sequence 1615 or annotated dense image sequence 1615a) and return a processed data set 1617 to the robot 100, e.g., to the controller 500 and/or web pad 310. The robot 100 may issue drive commands 1619 (e.g., via the controller 500 or web pad 310) to the drive system 200 based on the received processed data set 1617 for maneuvering about the scene 10.

[00218] After a threshold period of time or a threshold amount of image data 1601, 1603 is accumulated in the cloud storage 1622, the cloud service 1620 may execute one of a variety of off-line methods to process the image data set 1603 into a dense 3-D map or model 1605 of the scene 10 (environment) and then simplify this dense 3-D map or model 1605 into a 2-D height map 1607, which can be a 2D-map with height data at each point (e.g., similar to a 2-D topographical map). In some examples, the 2-D height map 1607 is a topographical map having X and Y coordinates with Z data. Each X,Y coordinate may have one or more Z points (i.e., height data). Unlike the dense 3-D map, which may have numerous Z points (e.g., hundreds or thousands of Z points) for each X,Y coordinate, the 2-D height map 1607 may have less than threshold number of Z points for each X,Y coordinate, such as between 2 and 20 (e.g., 10) points. A 2-D height map 1607 derived from a 3-D map of a table in a room may show a first Z point for the bottom surface of a table top and a second Z point for the top surface of the table top for each X,Y coordinate along the table. This information allows the robot 100 to determine if it can pass under the table top. By reducing the Z-points from a dense data set of a continuous range of Z points for each X, Y coordinate to a sparse data set of a select number of Z points indicative of a detected objects 12, the robot 100 can receive a 2-D height map 1607 having a relatively smaller size than the 3-D map used by the cloud service 1620. This, in turn, allows the robot 100 to store the 2-D height map 1607 on local memory having a practical and cost effective size as compared to the scalable memory space available to the cloud service 1620. The robot 100 receives the 2-D height map 1607 from the cloud 1620, which provides the robot 100 and associated controller 500 with navigational data for future work in the scene 10.

[00219] Additional methods and features of 3-D map data compression are disclosed in “Multi-Level Surface Maps For Outdoor Terrain Mapping and Loop Closing" by R. Triebel, P. Pfaff and W. Burgard; IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, which is hereby incorporated by reference in its entirety.

[00220] The cloud 1620 provides the robot 100 with on-demand scaling of resources (e.g., computational, processing, memory, etc.) that may not otherwise be practical or cost effective on the robot 100. For example, the cloud 1620 can provide scalable cloud storage 1622 that scales up to a first size for storing and/or processing a relatively large amount of data 1601, which may only be used for a short period of time and then discarded, and then scale back down to a second size. Moreover, the cloud 1620 can provide computer processing power for executing relatively complex computations or ‘brute force’ algorithms that might not otherwise be possible on the robot. By displacing computer processing power and memory to a scalable cloud 1620, the robot 100 can use a controller 500 having relatively less computing power and memory, thus providing a cost effective solution. Moreover, the robot 100 may execute real-time tasks (on the controller 500 or the web pad 310), such as obstacle avoidance, while passing non-real-time or non-time-sensitive tasks to the cloud 1620 for processing and later retrieval.

[00221] The cloud 1620 may execute one or more filters (e.g., a Bundle Adjustment, RANSAC, Expectation Maximization, SAM or other 3D structural estimation algorithms) for processing the image data set 1603 into a 3D representation. Once processed and a dense 3-D map 1605 has been created or updated, the image data set 1603 can be discarded from the cloud storage 1622, freeing up resources and allowing the cloud 1620 to scale accordingly. As a result, the robot 100 needs neither the on-board storage nor the processing to handle the storage and processing of the image data set 1603, due to the use of cloud based resources. The cloud 1620 may return processed navigational data 1601 or a map 1607 (e.g. a compressed 2-D height map) to the robot 100, which it can then use for relatively simpler localization and navigation processing.

[00222] Additional methods and features of 3-D reconstruction are disclosed in “3D Models From Extended Uncalibrated Video Sequences: Addressing Key-frame selection and projective drift" by J. Repko and M. Pollefeys; Fifth International Conference on 3-D

Digital Imaging and Modeling, 2005, which is hereby incorporated by reference in its entirety.

[00223] With respect to floor classification, the robot 100 may acquire images of the work surface 5 while maneuvering thereon about the scene 10. The controller 500 may receive the images and execute an object detection routine for object detection and obstacle avoidance (ODOA). In some implementations, the controller 500 associates information with the images (e.g., tags the images with data), such as accelerometer data traces, odometry data, and/or other data from the sensor system 400 along with timestamps. The images may capture drop offs, rug tassels, socks on rugs, etc. As in the previous example, the robot 100 can stream the image data 1601 up to the cloud storage 1622 while working for later batch processing. When the robot encounters an undesirable event (i.e. accidental collision) a special ‘hazard’ tag is inserted into the data set so that the data moments prior to the hazard may be identified for learning algorithms. Once the annotated image data set and associated tags 1603 are accumulated (potentially along with the data set from many other robots 100), a parallel cloud host 1620 may be launched to process the annotated image data set 1603 using a supervised learning algorithm, for example, that computes "hazard image" classes from the many images of a real environment that are temporally preceding hazard tags. Once the training of a hazard image class model 1609 is complete, the parameters for that model 1609 (a small amount of data) can be downloaded back down to many different robots 100. Thus, an entire fleet of robots 100 can adaptively learn elements of an environment on-line. Learning methods applicable to this method include genetic algorithms, neural networks, and support vector machines. All of these may be too complex and may take too much storage to run on-line (i.e., on a local robot processor) in a low-cost robot 100, but the cloud 1620 offers a robot fleet access to "fully trained" classifiers 1625.

[00224] A 'classifier' 1625 is a machine learning algorithm that usually employs an iterative training algorithm, to, e.g., minimize an error function, optimize/minimize a cost function, or otherwise improve in performance using training data. Typically the algorithm has a number of parameters whose values are learned from training data. Three types include supervised regression, supervised classification, and unsupervised classification via clustering or dimensionality reduction. Examples of classifiers 1625 are

Support Vector Machines (SVM) of various types and kernels (with gradient descent or other cost function minimizing technique), naive Bayes classifier, logistic regression, adaboost, K nearest neighbor ("K-NN") and/or K-NN regression, neural networks, random forests, and linear models. A 'classifier' classifies 'features', which may be represented by vectors, matrices, "descriptors", or other data sets, and identified by various algorithms, for example, histogram of oriented gradients (HOG), shape context histograms (SCH), color patches, texture patches, luminance patches, scale-invariant feature descriptors from SIFT, SURF, or the like, affine invariant detectors like MSER, labeled super pixels or segments within images, Hough transform or RANSAC line detection, and other features combinable herewith.

[00225] The classifiers 1625 can be executed on-line to help the robot systems avoid hazards in their environment just from a picture. Once the model parameters are determined the image data set 1603 stored in the cloud storage 1622 can be discarded. This example can be combined with the previous example, where determination of a 3-D structure of the environment allows identification of traversable regions for training the classification algorithm. One example learning technique is disclosed in “Long-Term Learning Using Multiple Models For Outdoor Autonomous Robot Navigation,” by Michael J. Procopio , Jane Mulligan , and Greg Grudic, 2007 IEEE International Conference on Intelligent Robots and Systems; which is hereby incorporated by reference in its entirety.

[00226] Streaming generally means transmitting an image sequence with some relation to an order and rate at which the image sequence was captured in real time, with or without buffering, whether or not packets are reordered between sender and receiver, as distinguished from dispatching a batch processing task (which can occur at very high data rates) or trickling images in a delayed manner. The term ‘streaming’ is distinguished from ‘dispatching a batch processing task’ to distinguish the rate at which different data is moved within and among entities of a robotic system, based on real-world bandwidth, processing and storage constraints.

[00227] Referring to FIGS. 16D and 16E, robots 100 at practical cost levels may have a limitation on robot bandwidth, computation and storage. By annotating the images 1611 with locally processed data 1613 (i.e. data from the robot sensor system 400), the robot 100 can send up images 1611, dense image sequences 1615, 1615a, or image data 1601 fairly infrequently (e.g., once every few seconds instead of many times each second), dramatically reducing an up-stream bandwidth as compared to continuous streaming of data. In addition, the controller 500 and/or web pad 310 may include a local cache (memory) where images 1611 and annotation data 1613 can be stored on the robot 100 when wireless connectivity is inaccessible, and transmitted when wireless connectivity is accessible. Buffering allows the robot 100 to collect a room worth of data, for example, in a room with bad signal reception and still transmit a complete data set 1601, 1603 once wireless connectivity is available again, such as in an adjacent room. Thus, data management on the robot 100 may be necessary for accommodating various communications environments.

[00228] As the mobile robot is piloted, directed, or autonomously navigates throughout a scene 10 (e.g., home, office, etc.), one or more cameras 320 and/or imaging sensors 450 may obtain one or more sequences 1615 of images 1611, each corresponding to a view obtained at a pose (position and orientation) of the robot 100 along the trajectory of the robot 100. A timer (e.g., of the controller 500) provides a reference time stamp for each image 1611, and the time stamp may be associated with annotations 1613 corresponding positional, localization, movement velocity or acceleration, sensor-based orientation, etc. At least some images 1611 are annotated with information, and all images 1611 may be annotated with time stamps or other metadata reflecting a robot status, image status, or the like. The images 1611 can be captured at some real-time capture rate, which need not be periodic (e.g., images 1611 may be captured at an adaptive rate, based on, e.g., processing availability, bandwidth availability, time of day, task, density of features of interest in current images, or the like).

[00229] The camera 320 and/or imaging sensor 450 may be in communication with a hardware encoder, a high-speed bus available to memory, an on-board processor, and storage (e.g., flash memory or a hard disk drive) for storing and accumulating captured images 1611 at a relatively high rate. One or more machine vision algorithms can be applied to the stored images 1611 to create models, such as a set of parameters for image classifiers 1625, that themselves do not consume more resources than are supportable on the robot 100. However, there are a few obstacles to overcome.

[00230] (1) Typically, identification of at least some training data (e.g., images 1611) or annotations 1613 describing features to be modeled should be applied before substantial image compression.

[00231] (2) Many algorithms and heuristics can be scaled and adapted to relatively large distributed compute infrastructures instanced in cloud services to complete modeling in a shorter period of time, but this requires getting the images 1611 to the target compute infrastructure across the public Internet.

[00232] (3) Bandwidth available for transporting relatively large data sets is limited by (i) the robot 100 being mobile with a wireless bandwidth typically occupied with other traffic and (ii) the Internet access may be bandwidth-limited in an upstream direction.

[00233] To overcome at least these obstacles, the dense images 1611 can be communicated (e.g., wirelessly) from the robot 100 to a local server 1640, which may be a wire network with storage at a local send rate that is relatively lower than a real-time capture rate. The local server 1640 can buffer and annotate the dense images 1611 with annotations 1613. Moreover, the local server 1640 may accumulate annotated dense image sequences 1615a for later communication to the cloud service 1620. The local server 1640 can communicate image data 1601 (e.g., dense images 1611, dense image sequences 1615, and/or annotated dense image sequences 1615a) to the cloud service 1620 at a cloud send rate that is slower than the real-time capture rate or at a rate suitable for servicing by a cloud computing infrastructure at high speed.

[00234] The cloud service 1620 can process the received image data 1601, e.g., by elastically dispatching sufficiently fast, parallel, and/or large image set classifier 1625 processing instances to train classifiers 1625 on the image data 1601 (e.g., on the dense images 1611 and associated annotations 1613) to provide a simplified data set 1617, derived from and representing the annotated dense image sequence 1615a, for example, but excluding any raw image data. The cloud service 1620 can transmit the simplified data set 1617 to the local server 1640, which communicates with the robot 100 or directly to the robot 100 after a processing interval, for example. Elements of the data set 1617 may be used by the robot 100 for issuing commands to the drive system 200 to maneuver the robot 100 with respect to the scene/environment 10. For example, a 'cord' or 'sock' or 'ingestible debris' classifier 1625 trained on the image data 1601 of one or many robots may be used with parameters returned to the robot 100 to identify image patterns and direct the robot 100 either away from a hazard or toward an area of interest.

[00235] When a wireless connection or consumer/commercial asymmetric broadband service is bandwidth limited in the direction of transmission, the dense image sequence 1615 can be buffered in a large storage device and uploaded over a period of time relatively longer than a collection time (i.e., a period of time to collect the dense image sequence 1615), taking advantage of packet switching, reordering, correction, quality of service, etc. For example, the upload may occur overnight, over a period of hours where the robot's trajectory was in minutes, over a period of days where the robot's trajectory was in hours, etc.

[00236] On the local server 1640 or the cloud services 1620, the dense images 1611 representing the sequence of views of the environment along a trajectory of the mobile robot 100 captured at a real-time capture rate can be received by a service 1623 related to a robot software platform of the robot 100. The service 1623, also referred to as a cloud gateway, may be an agent of the robot software platform or controlled independently by a third party. The cloud gateway 1623 may match the dense image sequence 1615 with an algorithm, classifier 1625, and application platform instance to be instantiated/provisioned as one or more virtual servers 1621 in the cloud service 1620 (e.g., depending on the annotation content) and packages the dense image sequence 1615 and any annotations 1613 as training data for the classifier 1625. Alternatively, the cloud gateway 1623 may handle the same information as prepackaged information from the robot 100, local base station, etc., and simply begin a dispatch of sufficient predetermined virtual server instances 1621, which elastically add other instances as compliant with an adaptive & elastic cloud services application programming interface (API) (e.g., Amazon EC2). The local server 1640, the cloud gateway 1623, a cloud services manager, or an initial virtual processor instance 1621 dispatches a batch processing task of reducing dense data within each of least some of the dense images 1611 in the received dense image sequence 1615, 1615a to a data set 1617 derived from and representing the dense image sequence 1615, 1615a. As necessary, new virtual processor instances 1621 can provisioned / instantiated (either all at once, or as the training task becomes more complex). The dense images 1611 and trained models 1609 can be kept in long term storage instances. The parameters of the trained classifier 1625 or model 1609 are returned to the robot 100 (e.g., directly or via an agent of the robot 100), excluding the sequence of raw images 1611.

[00237] FIG. 16F provides an exemplary arrangement 1600f of operations for a method of navigating the robot 100. The method includes capturing 1602f a streaming sequence 1615 of dense images 1611 of a scene 10 about the robot 100 along a locus of motion of the robot 100 at a real-time capture rate and associating 1604f annotations 1613 with at least some of the dense images 1611. The method also includes sending 1606f the dense images 1611 and annotations 1613 to a remote server 1620 at a send rate, which is slower than the real-time capture rate and receiving 1608f a data set 1607, 1617 from the remote server 1620 after a processing time interval. The data set 1607, 1617 is derived from and represents at least a portion of the dense image sequence 1615, 1615a and corresponding annotations 1613, but excludes raw image data of the sequence 1615, 1615a of dense images 1611. The method includes moving 1610f the robot 100 with respect to the scene 10 based on the received data set 1607, 1617.

[00238] The method may include sending the dense images 1611 and annotations 1613 to a local server and buffer 1640 (FIGS. 16D and 16E), and then sending the dense images 1611 and annotations 1613 to the remote server 1620 at a send rate slower than the real-time capture rate. The local server and buffer 1640 may be within a relatively short-range of the robot 100 (e.g., within between 20-100 feet or a wireless communication range). For example, the local and buffer 1640 may be a personal computer in user's home that houses the robot 100 or a local server of a building housing the robot 100.

[00239] In some implementations, the annotations 1613 include a time stamp, such as an absolute time reference corresponding to at least some of the dense images 1611, and pose-related sensor data, which may include at least one of odometry data, accelerometer data, tilt data, bump, and angular rate data. Annotations 1613 can be associated to the dense images 1611 that reflect hazard events captured in a time interval relative to a hazard response of the robot 100 (e.g., avoiding a cliff, escaping from a confining situation, etc.). In additional examples, associating 1604f annotations 1613 may include associating key-frame identifiers with a subset of the dense images 1611. The key-frame identifiers may allow identification of dense images 1611 based on properties of the key-frame identifiers (e.g., flag, type, group, moving, still, etc.).

[00240] The annotations 1613 may include a sparse set of 3-D points derived from structure and motion recovery of features tracked between dense images 1611 of the streaming sequence 1615 of dense images 1611. The sparse set of 3-D points may be from a volumetric point imaging device 450 on the robot 100. Moreover, the annotations 1613 may include camera parameters, such as a camera pose relative to individual 3-D points of the sparse set of 3-D points. Labels of traversable and non-traversable regions of the scene 10 may be annotations 1613 for the dense images 1611.

[00241] The data set 1607, 1617 may include one or more texture maps, such as the 2-D height map 1607, extracted from the dense images 1611 and/or a terrain map 1607 representing features within the dense images 1611 of the scene 10. The data set 1607, 1617 may include a trained classifier 1625 for classifying features within new dense images 1611 captured of the scene 10.

[00242] FIG. 16G provides an exemplary arrangement 1600g of operations for a method of abstracting mobile robot environmental data. The method includes receiving 1602g a sequence 1615 of dense images 1611 of a robot environment 10 from a mobile robot 100 at a receiving rate. The dense images 1611 are captured along a locus of motion of the mobile robot 100 at a real-time capture rate. The receiving rate is slower than the real-time capture rate. The method also includes receiving 1604g annotations 1613 associated with at least some of the dense images 1611 in the sequence 1615 of dense images 1611, and dispatching 1606g a batch processing task for reducing dense data within least some of the dense images 1611 to a data set 1607, 1617 representing at least a portion of the sequence 1615 of dense images 1611. The method also includes transmitting 1608g the data set 1617 to the mobile robot 100. The data set 1607, 1617 excludes raw image data of the sequence 1615 of dense images 1611.

[00243] In some implementations, the batch processing task includes processing the sequence 1615 of dense images 1611 into a dense 3-D model 1609 of the robot environment 10 and processing the dense 3-D model 1609 into a terrain model 1607 for a coordinate system of 2-D location and at least one height from a floor plane G. In some examples, the terrain model 1607 is for a coordinate system of 2-D location and a plurality of occupied and unoccupied height boundaries from a floor plane G. For example, a terrain model 1607 if room having a table would provide data indicating upper and lower heights of an associated table top, so that the robot 100 can determine if it can pass underneath the table.

[00244] The batch processing task may include accumulating dense image sequences 1615, 1615a corresponding to a plurality of robot environments 10 (e.g., so that the cloud 1620 can build classifiers 1625 for identifying features of interest in any environment). As such, the batch processing task may include a plurality of classifiers 1625 and/or training one or more classifiers 1625 on the sequence 1615 of dense images 1611. For example, the batch processing task may include associating annotations 1613 that reflect hazard events with dense images 1611 captured in a time interval relative to a hazard response of the mobile robot 100 and training a classifier 1625 of hazard-related dense images 1611 using the associated hazard event annotations 1613 and corresponding dense images 1611 as training data, e.g., to provide a data set 603, 1607, 1617 of model parameters for the classifier 1625. The classifier 1625 may include at least one Support Vector Machine that constructs at least one hyperplane for classification, and the model parameters define a trained hyperplane capable of classifying a data set 1603, 1607, 1617 into hazard-related classifications. The model parameters may include sufficient parameters to define a kernel of the Support Vector Machine and a soft margin parameter

[00245] In some examples, the batch processing task includes instantiating a scalable plurality of virtual processes 1621 proportionate to a scale of the dense image sequence 1615, 1615a to be processed. At least some of the virtual processes 1621 are released after transmission of the data set 1607, 1617 to the robot 100. Similarly, the batch processing task may include instantiating a scalable plurality of virtual storage 1622 proportionate to a scale of the dense image sequence 1615, 1615a to be stored. At least some of virtual storage 1622 is released after transmission of the data set 1607, 1617 to the robot 100. Moreover, the batch processing task may include distributing a scalable plurality of virtual servers 1621 according to one of geographic proximity to the mobile robot 100 and/or network traffic from a plurality of mobile robots 100.

[00246] U.S. Patent Publication 2011/0238857, "Committed Processing Rates for Shared Resources", Certain et al., published Sept. 29, 2011, is hereby incorporated by reference in its entirety. The described cloud processing infrastructure in Certain et al. is one species combinable herewith, for example, management system 202 or node manager module 108 being species or portions of the cloud gateway 1623, program execution service (PES) and/or virtual machines 110 being species or portions of elastic processing servers 1621, and archival storage 222 or block data service (BDS) 204 or archival manager 224 being species or portions of long term storage instance 1622.

[00247] FIG. 16H is a schematic view of an exemplary mobile human interface robot system architecture 1600d. In the example shown, application developers 1602 can access and use application development tools 1640 to produce applications 1610 executable on the web pad 310 or a computing device 1604 (e.g., desktop computer, tablet computer, mobile device, etc.) in communication with the cloud 1620. Exemplary application development tools 1640 may include, but are not limited to, an integrated development environment 1642, a software development kit (SDK) libraries 1644, development or SDK tools 1646 (e.g., modules of software code, a simulator, a cloud usage monitor and service configurator, and a cloud services extension uploader/deployer), and/or source code 1648. The SDK libraries 1644 may allow enterprise developers 1602 to leverage mapping, navigation, scheduling and conferencing technologies of the robot 100 in the applications 1610. Exemplary applications 1610 may include, but are not limited to, a map builder 1610a, a mapping and navigation application 1610b, a video conferencing application 1610c, a scheduling application 1610d, and a usage application 1610e. The applications 1610 may be stored on one or more applications servers 1650 (e.g., cloud storage 1622) in the cloud 1620 and can be accessed through a cloud services application programming interface (API). The cloud 1620 may include one or more databases 1660 and a simulator 1670. A web services API can allow communication between the robot 100 and the cloud 1620 (e.g., and the application server(s) 1650, database(s) 1660, and the simulator 1670). External systems 1680 may interact with the cloud 1620 as well, for example, to access the applications 1610.

[00248] In some examples, the map builder application 1610a can build a map 1700 (FIG. 17A) of an environment around the robot 100 by linking together pictures or video captured by the camera 320 or 3-D imaging sensor 450 using reference coordinates, as provided by odometry, a global positioning system, and/or way-point navigation. The map may provide an indoor or outside street or path view of the environment. For malls or shopping centers, the map can provide a path tour through-out the mall with each store marked as a reference location with additional linked images or video and/or promotional information. The map and/or constituent images or video can be stored in the database 1660.

[00249] The applications 1610 may seamlessly communicate with the cloud services, which may be customized and extended based on the needs of each user entity. Enterprise developers 1602 may upload cloud-side extensions to the cloud 1620 that fetch data from external proprietary systems for use by an application 1610. The simulator 1670 allows the developers 1602 to build enterprise-scale applications without the robot 100 or associated robot hardware. Users may use the SDK tools 1646 (e.g., usage monitor and service configurator) to add or disable cloud services.

[00250] Referring to FIGS. 17A and 17B, in some circumstances, the robot 100 receives an occupancy map 1700 of objects 12 in a scene 10 and/or work area 5, or the robot controller 500 produces (and may update) the occupancy map 1700 based on image data and/or image depth data received from an imaging sensor 450 (e.g., the second 3-D image sensor 450b) over time. Simultaneous localization and mapping (SLAM) is a technique that may be used by the robot 100 to build up a map 1700 within an unknown environment or scene 10 (without a priori knowledge), or to update a map 1700 within a known environment (with a priori knowledge from a given map), while at the same time keeping track of its current location. Maps 1700 can be used to determine a location within an environment and to depict an environment for planning and navigation. The maps 1700 support the assessment of actual location by recording information obtained from a form of perception and comparing it to a current set of perceptions. The benefit of a map 1700 in aiding the assessment of a location increases as the precision and quality of the current perceptions decrease. Maps 1700 generally represent the state at the time that the map 1700 is provided or produced. This is not necessarily consistent with the state of the environment at the time the map 1700 is used. Other localization techniques include monocular visual SLAM (MonoSLAM) and implementations using an extended Kalman filter (EKF) for MonoSLAM solutions.

[00251] The controller 500 may execute a scale-invariant feature transform (SIFT) to detect and describe local features in captured images. For any object 12 in an image, interesting points on the object 12 can be extracted to provide a "feature description" of the object 12. This description, extracted from a training image, can then be used to identify the object 12 when attempting to locate the object 12 in a test image containing many other objects. To perform reliable recognition, it is important that the features extracted from the training image be detectable even under changes in image scale, noise and illumination. Such points usually lie on high-contrast regions of the image, such as object edges. For object recognition and detection, the robot 100 may use a SIFT to find distinctive key points that are invariant to location, scale and rotation, and robust to affine transformations (changes in scale, rotation, shear, and position) and changes in illumination. In some implementations, the robot 100 captures multiple images (using the camera 320 and/or imaging sensor 450) of a scene 10 or object 12 (e.g., under different conditions, from different angles, etc.) and stores the images, such as in a matrix. The robot 100 can access the stored images to identify a new image by comparison, filter, etc. For example, SIFT features can be obtained from an input image and matched to a SIFT feature database obtained from training images (captured previously). The feature matching can be done through a Euclidean-distance based nearest neighbor approach. A Hough transform may be used to increase object identification by clustering those features that belong to the same object and reject the matches that are left out in the clustering process. SURF (Speeded Up Robust Feature) may be a robust image detector & descriptor.

[00252] In addition to localization of the robot 100 in the scene 10 (e.g., the environment about the robot 100), the robot 100 may travel to other points in a connected space (e.g., the work area 5) using the sensor system 400. The robot 100 may include a short range type of imaging sensor 450a (e.g., mounted on the underside of the torso 140, as shown in FIGS. 1 and 3) for mapping a nearby area about the robot 110 and discerning relatively close objects 12, and a long range type of imaging sensor 450b (e.g., mounted on the head 160, as shown in FIGS. 1 and 3) for mapping a relatively larger area about the robot 100 and discerning relatively far away objects 12. The robot 100 can use the occupancy map 1700 to identify known objects 12 in the scene 10 as well as occlusions

16 (e.g., where an object 12 should or should not be, but cannot be confirmed from the current vantage point). The robot 100 can register an occlusion 16 or new object 12 in the scene 10 and attempt to circumnavigate the occlusion 16 or new object 12 to verify the location of new object 12 or any objects 12 in the occlusion 16. Moreover, using the occupancy map 1700, the robot 100 can determine and track movement of an object 12 in the scene 10. For example, the imaging sensor 450, 450a, 450b may detect a new position 12' of the object 12 in the scene 10 while not detecting a mapped position of the object 12 in the scene 10. The robot 100 can register the position of the old object 12 as an occlusion 16 and try to circumnavigate the occlusion 16 to verify the location of the object 12. The robot 100 may compare new image depth data with previous image depth data (e.g., the map 1700) and assign a confidence level of the location of the object 12 in the scene 10. The location confidence level of objects 12 within the scene 10 can time out after a threshold period of time. The sensor system 400 can update location confidence levels of each object 12 after each imaging cycle of the sensor system 400. In some examples, a detected new occlusion 16 (e.g., a missing object 12 from the occupancy map 1700) within an occlusion detection period (e.g., less than ten seconds), may signify a "live" object 12 (e.g., a moving object 12) in the scene 10.

[00253] In some implementations, a second object 12b of interest, located behind a detected first object 12a in the scene 10, may be initially undetected as an occlusion 16 in the scene 10. An occlusion 16 can be area in the scene 10 that is not readily detectable or viewable by the imaging sensor 450, 450a, 450b. In the example shown, the sensor system 400 (e.g., or a portion thereof, such as imaging sensor 450, 450a, 450b) of the robot 100 has a field of view 452 with a viewing angle θ_V (which can be any angle between 0 degrees and 360 degrees) to view the scene 10. In some examples, the imaging sensor 450 includes omni-directional optics for a 360 degree viewing angle θ_V; while in other examples, the imaging sensor 450, 450a, 450b has a viewing angle θ_V of less than 360 degrees (e.g., between about 45 degrees and 180 degrees). In examples, where the viewing angle θ_V is less than 360 degrees, the imaging sensor 450, 450a, 450b (or components thereof) may rotate with respect to the robot body 110 to achieve a viewing angle θ_V of 360 degrees. In some implementations, the imaging sensor 450, 450a, 450b or portions thereof, can move with respect to the robot body 110 and/or drive system 200. Moreover, in order to detect the second object 12b, the robot 100 may move the imaging sensor 450, 450a, 450b by driving about the scene 10 in one or more directions (e.g., by translating and/or rotating on the work surface 5) to obtain a vantage point that allows detection of the second object 10b. Robot movement or independent movement of the imaging sensor 450, 450a, 450b, or portions thereof, may resolve monocular difficulties as well.

[00254] A confidence level may be assigned to detected locations or tracked movements of objects 12 in the working area 5. For example, upon producing or updating the occupancy map 1700, the controller 500 may assign a confidence level for each object 12 on the map 1700. The confidence level can be directly proportional to a probability that the object 12 actually located in the working area 5 as indicated on the map 1700. The confidence level may be determined by a number of factors, such as the number and type of sensors used to detect the object 12. For example, the contact sensor 430 may provide the highest level of confidence, as the contact sensor 430 senses actual contact with the object 12 by the robot 100. The imaging sensor 450 may provide a different level of confidence, which may be higher than the proximity sensor 430. Data received from more than one sensor of the sensor system 400 can be aggregated or accumulated for providing a relatively higher level of confidence over any single sensor.

[00255] Odometry is the use of data from the movement of actuators to estimate change in position over time (distance traveled). In some examples, an encoder is disposed on the drive system 200 for measuring wheel revolutions, therefore a distance traveled by the robot 100. The controller 500 may use odometry in assessing a confidence level for an object location. In some implementations, the sensor system 400 includes an odometer and/or an angular rate sensor (e.g., gyroscope or the IMU 470) for sensing a distance traveled by the robot 100. A gyroscope is a device for measuring or maintaining orientation, based on the principles of conservation of angular momentum. The controller 500 may use odometry and/or gyro signals received from the odometer and/or angular rate sensor, respectively, to determine a location of the robot 100 in a working area 5 and/or on an occupancy map 1700. In some examples, the controller 500 uses dead reckoning. Dead reckoning is the process of estimating a current position based upon a previously determined position, and advancing that position based upon known or estimated speeds over elapsed time, and course. By knowing a robot location in the working area 5 (e.g., via odometry, gyroscope, etc.) as well as a sensed location of one or more objects 12 in the working area 5 (via the sensor system 400), the controller 500 can assess a relatively higher confidence level of a location or movement of an object 12 on the occupancy map 1700 and in the working area 5 (versus without the use of odometry or a gyroscope).

[00256] Odometry based on wheel motion can be electrically noisy. The controller 500 may receive image data from the imaging sensor 450 of the environment or scene 10 about the robot 100 for computing robot motion, independently of wheel based odometry of the drive system 200, through visual odometry. Visual odometry may entail using optical flow to determine the motion of the imaging sensor 450. The controller 500 can use the calculated motion based on imaging data of the imaging sensor 450 for correcting any errors in the wheel based odometry, thus allowing for improved mapping and motion control. Visual odometry may have limitations with low-texture or low-light scenes 10, if the imaging sensor 450 cannot track features within the captured image(s).

[00257] Other details and features on odometry and imaging systems, which may combinable with those described herein, can be found in U.S. Patent 7,158,317 (describing a "depth-of field" imaging system), and U.S. Patent 7,115,849 (describing wavefront coding interference contrast imaging systems), the contents of which are hereby incorporated by reference in their entireties.

[00258] When a robot is new to a building that it will be working in, the robot may need to be shown around or provided with a map of the building (e.g., room and hallway locations) for autonomous navigation. For example, in a hospital, the robot may need to know the location of each patient room, nursing stations, etc. In some implementations, the robot 100 receives a layout map 1810, such as the one shown in FIG. 18A, and can be trained to learn the layout map 1810. For example, while leading the robot 100 around the building, the robot 100 may record specific locations corresponding to locations on the layout map 1810. The robot 100 may display the layout map 1810 on the web pad 310 and when the user takes the robot 100 to a specific location, the user can tag that location on the layout map 1810 (e.g., using a touch screen or other pointing device of the web pads 310). The user may choose to enter a label for a tagged location, like a room name or a room number. At the time of tagging, the robot 100 may store the tag, with a point on the layout map 1810 and a corresponding point on a robot map 1820, such as the one shown in FIG. 18B.

[00259] Using the sensor system 400, the robot 100 may build the robot map 1820 as it moves around. For example, the sensor system 400 can provide information on how far the robot 100 has moved and a direction of travel. The robot map 1820 may include fixed obstacles in addition to the walls provided in the layout map 1810. The robot 100 may use the robot map 1820 to execute autonomous navigation. In the robot map at 1820, the "walls" may not look perfectly straight, for example, due to detected packing creates along the wall in the corresponding hallway and/or furniture detected inside various cubicles. Moreover, rotational and resolution differences may exist between the layout map 1810 and the robot map 1820.

[00260] After map training, when a user wants to send the robot 100 to a location, the user can either refer to a label/tag (e.g., enter a label or tag into a location text box displayed on the web pad 310) or the robot 100 can display the layout map 1810 to the user on the web pad 310 and the user may select the location on the layout map 1810. If the user selects a tagged layout map location, the robot 100 can easily determine the location on the robot map 1820 that corresponds to the selected location on the layout map 1810 and can proceed to navigate to the selected location.

[00261] If the selected location on the layout map 1810 is not a tagged location, the robot 100 determines a corresponding location on the robot map 1820. In some implementations, the robot 100 computes a scaling size, origin mapping, and rotation between the layout map 1810 and the robot map 1820 using existing tagged locations, and then applies the computed parameters to determine the robot map location (e.g., using an affine transformation or coordinates).

[00262] The robot map 1820 may not be the same orientation and scale as the layout map 1810. Moreover, the layout map may not be to scale and may have distortions that vary by map area. For example, a layout map 1810 created by scanning a fire evacuation map typically seen in hotels, offices, and hospitals is usually not to drawn scale and can even have different scales in different regions of the map. The robot map 1820 may have its own distortions. For example, locations on the robot map 1820 may been computed by counting wheel turns as a measure of distance, and if the floor was slightly slippery or turning of corners caused extra wheel, inaccurate rotation calculations may cause the robot 100 to determine inaccurate locations of mapped objects.

[00263] A method of mapping a given point 1814 on the layout map 1810 to a corresponding point 1824 on the robot map 1820 may include using existing tagged 1812 points to compute a local distortion between the layout map 1810 and the robot map 1820 in a region (e.g., within a threshold radius) containing the layout map point. The method further includes applying a distortion calculation to the layout map point 1814 in order to find a corresponding robot map point 1824. The reverse can be done if you are starting with a given point on the robot map 1820 and want to find a corresponding point on the layout map 1810, for example, for asking the robot for its current location.

[00264] FIG. 18C provide an exemplary arrangement 1800 of operations for operating the robot 100 to navigate about an environment using the layout map 1810 and the robot map 1820. With reference to FIGS. 18B and 18C, the operations include receiving 1802c a layout map 1810 corresponding to an environment of the robot 100, moving 1804c the robot 100 in the environment to a layout map location 1812 on the layout map 1810, recording 1806c a robot map location 1822 on a robot map 1820 corresponding to the environment and produced by the robot 100, determining 1808c a distortion between the robot map 1820 and the layout map 1810 using the recorded robot map locations 1822 and the corresponding layout map locations 1812, and applying 1810c the determined distortion to a target layout map location 1814 to determine a corresponding target robot map location 1824, thus allowing the robot to navigate to the selected location 1814 on the layout map 1810. In some implementations it operations include determining a scaling size, origin mapping, and rotation between the layout map and the robot map using existing tagged locations and resolving a robot map location corresponding to the selected target layout map location 1814. The operations may include applying an affine transformation to the determined scaling size, origin mapping, and rotation to resolve the robot map location.

[00265] Referring to FIGS. 19A-19C, in some implementations, the method includes using tagged layout map points 1912 (also referred to recorded layout map locations) to derive a triangulation of an area inside a bounding shape containing the tagged layout map points 1912, such that all areas of the layout map 1810 are covered by at least one triangle 1910 whose vertices are at a tagged layout map points 1912. The method further includes finding the triangle 1910 that contains the selected layout map point 1914 and determining a scale, rotation, translation, and skew between the triangle 1910 mapped in the layout map 1810 and a corresponding triangle 1920 mapped in the robot map 1820 (i.e., the robot map triangle with the same tagged vertices). The method includes applying the determined scale, rotation, translation, and skew to the selected layout map point 1914 in order to find a corresponding robot map point 1924.

[00266] FIG. 19C provide an exemplary arrangement 1900 of operations for determining the target robot map location 1924. The operations include determining 1902 a triangulation between layout map locations that bound the target layout map location, determining 1904 a scale, rotation, translation, and skew between a triangle mapped in the layout map and a corresponding triangle mapped in the robot map and applying 1906 the determined scale, rotation, translation, and skew to the target layout map location to determine the corresponding robot map point.

[00267] Referring to FIGS. 20A and 20B, in another example, the method includes determining the distances of all tagged points 1912 in the layout map 1810 to the selected layout map point 1914 and determining a centroid 2012 of the layout map tagged points 1912. The method also includes determining a centroid 2022 of all tagged points 1922 on the robot map 1820. For each tagged layout map point 1912, the method includes determining a rotation and a length scaling needed to transform a vector 2014 that runs from the layout map centroid 2012 to the selected layout point 1914 into a vector 2024 that runs from the robot map centroid 2022 to the robot map point 1924. Using this data, the method further includes determining an average rotation and scale. For each tagged layout map point 1912, the method further includes determining an “ideal robot map coordinate” point 1924i by applying the centroid translations, the average rotation, and the average scale to the selected layout map point 1914. Moreover, for each tagged layout map point 1912, the method includes determining a distance from that layout map point 1912 to the selected layout map point 1914 and sorting the tagged layout map points 1912 by these distances, shortest distance to longest. The method includes determining an “influence factor” for each tagged layout map point 1912, using either the inverse square of the distance between each tagged layout map point 1912 and the selected layout map point 1914. Then for each tagged layout map point 1912, the method includes determining a vector which is the difference between the "ideal robot map coordinate" point 1924i and robot map point 1924, prorated by using the influence factors of the tagged layout map points 1912. The method includes summing the prorated vectors and adding them to "ideal robot map coordinate" point 1924i for the selected layout map point 1914. The result is the corresponding robot map point 1924 on the robot map 1820. In some examples, this method/algorithm includes only the closest N tagged layout map point 1912 rather than all tagged layout map point 1912.

[00268] FIG. 20C provide an exemplary arrangement 2000 of operations for determining a target robot map location 1924 using the layout map 1810 and the robot map 1820. The operations include determining 2002 distances between all layout map locations and the target layout map location, determining 2004 a centroid of the layout map locations, determining 2006 a centroid of all recorded robot map locations, and for each layout map location, determining 2006 a rotation and a length scaling to transform a vector running from the layout map centroid to the target layout location into a vector running from the robot map centroid to the target robot map location.

[00269] Referring again to FIG. 16C, although the robot 100 may operate autonomously, a user may wish to control or manage the robot 100 through an application 1610. For example, a user may wish to control or define movement of the robot 100 within an environment or scene 10, as by providing navigation points on a map. In some implementations, the map builder application 1610a allows the user to create an occupancy map 1700 (FIG. 17A) or a layout map 1810 (FIG. 18A) of a scene 10 based on sensor data generated by the sensor system 400 of one or more robots 100 in the scene 10. In some examples, robot maps 1810 can be post-processed to produce one or more vector-based layout maps 1810. The map builder application 1610a may allow the user to customize the maps 1700, 1810 (e.g., lining up walls that should look parallel, etc). Moreover, annotations recognizable by the robot 100 and by the application 1610a (and/or other applications 1610) can be added to the maps 1700, 1810. In some implementations, the map builder application 1610a may store robot map data, layout map data, user-defined objects, and annotations securely in the cloud storage 1622 on the cloud 1620, using a cloud service. Relevant data sets may be pushed from the cloud services to the appropriate robots 100 and applications 1610.

[00270] In some implementations, the mapping and navigation application 1610b (FIG. 16C) allows users to specifying a destination location on a layout map 1810 and request the robot 100 to drive to the destination location. For example, a user may execute the mapping and navigation application 1610b on a computing device, such as a computer, tablet computer, mobile device, etc, which is in communication with the cloud 1620. The user can access a layout map 1810 of the environment about the robot 100, mark or otherwise set a destination location on the layout map 1810, and request the robot 100 to move to the destination location. The robot 100 may then autonomously navigate to the destination location using the layout map 1810 and/or a corresponding robot map 1820. To navigate to the destination location, the robot 100 may rely on its ability to discern its local perceptual space (i.e., the space around the robot 100 as perceived through the sensor system 400) and execute an object detection obstacle avoidance (ODOA) strategy.

[00271] Referring to FIGS. 11B and 21A-21D, in some implementations, the robot 100 (e.g., the control system 510 shown in FIG. 13) classifies its local perceptual space into three categories: obstacles (black) 2102, unknown (gray) 2104, and known free (white) 2106. Obstacles 2102 are observed (i.e., sensed) points above the ground G that are below a height of the robot 100 and observed points below the ground G (e.g., holes, steps down, etc.). Known free 2106 corresponds to areas where the 3-D image sensors 450 can see the ground G. Data from all sensors in the sensor system 400 can be combined into a discretized 3-D voxel grid. The 3-D grid can then be analyzed and converted into a 2-D grid 2100 with the three local perceptual space classifications. FIG. 21A provides an exemplary schematic view of the local perceptual space of the robot 100 while stationary. The information in the 3-D voxel grid has persistence, but decays over time if it is not reinforced. When the robot 100 is moving, it has more known free area 2106 to navigate in because of persistence.

[00272] An object detection obstacle avoidance (ODOA) navigation strategy for the control system 510 may include either accepting or rejecting potential robot positions that would result from commands. Potential robot paths 2110 can be generated many levels deep with different commands and resulting robot positions at each level. FIG. 21B provides an exemplary schematic view of the local perceptual space of the robot 100 while moving. An ODOA behavior 600b (FIG. 13) can evaluate each predicted robot path 2110. These evaluations can be used by the action selection engine 580 to determine a preferred outcome and a corresponding robot command. For example, for each robot position 2120 in the robot path 2110, the ODOA behavior 600b can execute a method for object detection and obstacle avoidance that includes identifying each cell in the grid 2100 that is in a bounding box around a corresponding position of the robot 100, receiving a classification of each cell. For each cell classified as an obstacle or unknown, retrieving a grid point corresponding to the cell and executing a collision check by determining if the grid point is within a collision circle about a location of the robot 100. If the grid point is within the collision circle, the method further includes executing a triangle test of whether the grid point is within a collision triangle (e.g., the robot 100 can be modeled as triangle). If the grid point is within the collision triangle, the method includes rejecting the grid point. If the robot position is inside of a sensor system field of view of parent grid points on the robot path 2110, then the "unknown" grid points are ignored because it is assumed that by the time the robot 100 reaches those grid points, it will be known.

[00273] The method may include determining whether any obstacle collisions are present within a robot path area (e.g., as modeled by a rectangle) between successive robot positions 2120 in the robot path 2110, to prevent robot collisions during the transition from one robot position 2120 to the next.

[00274] FIG. 21C provides a schematic view of the local perceptual space of the robot 100 and a sensor system field of view 405 (the control system 510 may use only certain sensor, such as the first and second 3-D image sensors 450a, 450b, for robot path determination). Taking advantage of the holonomic mobility of the drive system 200, the robot 100 can use the persistence of the known ground G to allow it to drive in directions where the sensor system field of view 405 does not actively cover. For example, if the robot 100 has been sitting still with the first and second 3-D image sensors 450a, 450b pointing forward, although the robot 100 is capable of driving sideways, the control system 510 will reject the proposed move, because the robot 100 does not know what is to its side, as illustrated in the example shown in FIG. 21C, which shows an unknown classified area to the side of the robot 100. If the robot 100 is driving forward with the first and second 3-D image sensors 450a, 450b pointing forward, then the ground G next to the robot 100 may be classified as known free 2106, because both the first and second 3-D image sensors 450a, 450b can view the ground G as free as the robot 100 drives forward and persistence of the classification has not decayed yet. (See e.g., FIG. 21B.) In such situations the robot 100 can drive sideways.

[00275] Referring to FIG. 21D, in some examples, given a large number of possible trajectories with holonomic mobility, the ODOA behavior 600b may cause robot to choose trajectories where it will (although not currently) see where it is going. For example, the robot 100 can anticipate the sensor field of view orientations that will allow the control system 510 to detect objects. Since the robot can rotate while translating, the robot can increase the sensor field of view 405 while driving.

[00276] By understanding the field of view 405 of the sensor system 400 and what it will see at different positions, the robot 100 can select movement trajectories that help it to see where it is going. For example, when turning a corner, the robot 100 may reject trajectories that make a hard turn around the corner because the robot 100 may end up in a robot position 2120 that is not sensor system field of view 405 of a parent robot position 2120 and of which it currently has no knowledge of, as shown in FIG. 21E. Instead, the robot 100 may select a movement trajectory that turns to face a desired direction of motion early and use the holonomic mobility of the drive system 200 to move sideways and then straight around the corner, as shown in FIG. 21F.

[00277] In some examples, the mapping and navigation application 1610b (FIG. 16C) provides teleoperation functionality. For example, the user can drive the robot 100 using video waypoint driving (e.g., using one or more of the cameras or imaging sensors 320, 450). The user may alter a height H_L of the leg 130 to raise/lower the height H_T of the torso 140 (FIG. 14) to alter a view field of one of the imaging sensors 450 and/or pan and/or tilt the robot head 160 to alter a view field of a supported camera 320 or imaging sensor 450, 450b (see e.g., FIGS 11A and 11B). Moreover, the user can rotate the robot about its Z-axis using the drive system 200 to gain other fields of view of the cameras or imaging sensors 320, 450. In some examples, the mapping and navigation application

1610b allows the user to switch between multiple layout maps 1810 (e.g., for different environments or different robots 100) and/or manage multiple robots 100 on one layout map 1810. The mapping and navigation application 1610b may communicate with the cloud services API to enforce policies on proper robot usage set forth by owners or organizations of the robots 100.

[00278] Referring again to FIG. 16C, in some implementations, the video conferencing application 1610c allows a user to initiate and/or participate in a video conferencing session with other users. In some examples, the video conferencing application 1610c allows a user to initiate and/or participate in a video conferencing session with a user of the robot 100, a remote user on a computing device connected to the cloud 1620 and/or another remote user connected to the Internet using a mobile handheld device. The video conferencing application 1610c may provide an electronic whiteboard for sharing information, an image viewer, and/or a PDF viewer.

[00279] The scheduling application 1610d allows users to schedule usage of one or more robots 100. When there are fewer robots 100 than the people who want to use them, the robots 100 become scarce resources and scheduling may be needed. Scheduling resolves conflicts in resource allocations and enables higher resource utilization. The scheduling application 1610d can be robot-centric and may integrate with third party calendaring systems, such as Microsoft Outlook or Google Calendar. In some examples, the scheduling application 1610d communicates with the cloud 1620 through one or more cloud services to dispatch robots 100 at pre-scheduled times. The scheduling application 1610d may integrate time-related data (e.g., maintenance schedule, etc.) with other robot data (e.g., robot locations, health status, etc.) to allow selection of a robot 100 by the cloud services for missions specified by the user.

[00280] In one scenario, a doctor may access the scheduling application 1610d on a computing device (e.g., a portable tablet computer or hand held device) in communication with the cloud 1620 for scheduling rounds at a remote hospital later in the week. The scheduling application 1610d can schedule robots 100 in a similar manner to allocating a conference room on a electronic calendar. The cloud services manage the schedules. If in the middle of the night, the doctor gets a call that a critical patient at a remote hospital needs to be seen, the doctor can request a robot 100 using the scheduling application 1610d and/or send a robot 100 to a patient room using the mapping and navigation application 1610b. The doctor may access medical records on his computing device (e.g., by accessing the cloud storage 1622) and video or imagery of the patient using the video conferencing application 1610c. The cloud services may integrate with robot management, an electronic health record systems and medical imaging systems. The doctor may control movement of the robot 100 remotely to interact with the patient. If the patent speaks only Portuguese, the video conferencing application 1610c may automatically translate languages or a 3rd party translator may join the video conference using another computing device in communication with the cloud 1620 (e.g., via the Internet). The translation services can be requested, fulfilled, recorded, and billed using the cloud services.

[00281] The usage / statistics application 1610e can be a general-purpose application for users to monitor robot usage, produce robot usage reports, and/or manage a fleet of robots 100. This application 1610e may also provide general operating and troubleshooting information for the robot 100. In some examples, the usage / statistics application 1610e allows the user to add/disable services associated with use of the robot 100, register for use of one or more simulators 1670, modify usage policies on the robot, etc.

[00282] In another scenario, a business may have a fleet of robots 100 for at least one telepresence application. A location manager may monitor a status of one or more robots 100 (e.g., location, usage and maintenance schedules, battery info, location history, etc.) using the usage / statistics application 1610e executing on a computing device in communication with the cloud 1620 (e.g., via the Internet). In some examples, the location manager can assist a user with a robot issue by sharing a user session. The location manager can issue commands to any of the robots 100 using an application 1610 to navigate the corresponding robot 100, speak through the robot 100 (i.e., telepresence), enter into a power-saving mode (e.g., reduce functionality), find a charger, etc. The location manager or a user can use applications 1610 to manage users, security, layout maps 1810, video view fields, add/remove robots to/from the fleet, and more. Remote operators of the robot 100 can schedule/reschedule/cancel a robot appointment (e.g., using the scheduling application 1610d) and attend a training course using a simulated robot that roams a simulated space (e.g., using the simulator 1670 executing on a cloud server).

[00283] The SDK libraries 1644 may include one or more source code libraries for use by developers 1602 of applications 1610. For example, a visual component library can provide graphical user interface or visual components having interfaces for accessing encapsulated functionality. Exemplary visual components include code classes for drawing layout map tiles and robots, video conferencing, viewing images and documents, and/or displaying calendars or schedules. A robot communication library (e.g., a web services API) can provide a RESTful (Representational State Transfer), JSON (JavaScript Object Notation)-based API for communicating directly with the robot 100. The robot communication library can offer Objective-C binding (e.g., for iOS development) and Java binding (e.g., for Android development). These object-oriented APIs allow applications 1610 to communicate with the robot 100, while encapsulating from the developers 1602 underlying data transfer protocol(s) of the robot 100. A person following routine of the robot communication library may return a video screen coordinate corresponding to a person tracked by the robot 100. A facial recognition routine of the , robot communication library may return a coordinate of a face on a camera view of the camera 320 and optionally the name of the recognized tracked person. Table 1 provides an exemplary list of robot communication services.

Service	Description
Database Service	List all available map databases on the robot 100.
Create Database Service	Create a robot map database.
Delete Database Service	Delete a robot map database.
Map List Service	Return a list of maps from a map database on the robot 100.
Map Service	Return a specific robot map.
Create Map Service	Create a robot map in a database.
Delete Map Service	Delete a robot map from a database.
Create Tag Service	Create a tag for a map in a database (e.g., by providing x, y, and z coordinates of the tag position an orientation angle of the robot, in radians, and a brief description of the tag).
Delete Tag Service	Delete a tag for a map in a database.
List Tags Service	List tags for a specified map in a specified database.
Cameras Service	List available cameras 320, 450 on the robot 100.
Camera Image Service	Take a snapshot from a camera 320, 450 on the robot 100.
Robot Position Service	Return a current position of the robot 100. The position can be returned as: • x -- a distance along an x-axis from an origin (in meters). • y -- a distance along a y-axis from an origin (in meters). • theta -- an angle from the x-axis, measured counterclockwise in radians.
Robot Destination Service	Sets a destination location of the robot 100 and commands the robot 100 to begin moving to that location.
Drive-To-Tag Service	Drives the robot 100 to a tagged destination in a map.
Stop Robot Service	Commands the robot 100 to stop moving.
Robot Info Service	Provide basic robot information (e.g., returns a dictionary of the robot information).

Table 1

[00284] A cloud services communication library may include APIs that allow applications 1610 to communicate with the cloud 1620 (e.g., with cloud storage 1622, applications servers 1650, databases 1660 and the simulator 1670) and/or robots 100 in communication with the cloud 1620. The cloud services communication library can be provided in both Objective-C and Java bindings. Examples of cloud services APIs include a navigation API (e.g., to retrieve positions, set destinations, etc.), a map storage and retrieval PAI, a camera feed API, a teleoperation API, a usage statistics API, and others.

[00285] A cloud services extensibility interface may allow the cloud services to interact with web services from external sources. For example, the cloud services may define a set of extension interfaces that allow enterprise developers 1602 to implement interfaces for external proprietary systems. The extensions can be uploaded and deployed to the cloud infrastructure. In some examples, the cloud services can adopt standard extensibility interface defined by various industry consortiums.

[00286] The simulator 1670 may allow debugging and testing of applications 1610 without connectivity to the robot 100. The simulator 1670 can model or simulate operation of the robot 100 without actually communicating with the robot 100 (e.g., for path planning and accessing map databases). For executing simulations, in some implementations, the simulator 1670 produces a map database (e.g., from a layout map 1810) without using the robot 100. This may involve image processing (e.g., edge detection) so that features (like walls, corners, columns, etc) are automatically identified. The simulator 1670 can use the map database to simulate path planning in an environment dictated by the layout map 1810.

[00287] A cloud services extension uploader/deployer may allow users upload extensions to the cloud 1620, connect to external third party user authentication systems, access external databases or storage (e.g., patient info for pre-consult and post-consult), access images for illustration in video conferencing sessions, etc. The cloud service extension interface may allow integration of proprietary systems with the cloud 1620.

[00288] Various implementations of the systems and techniques described here can be realized in digital electronic circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These various implementations can include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device.

[00289] These computer programs (also known as programs, software, software applications or code) include machine instructions for a programmable processor, and can be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the terms "machine-readable medium" and "computer-readable medium" refer to any computer program product, apparatus and/or device (e.g., magnetic discs, optical disks, memory, Programmable Logic Devices (PLDs)) used to provide machine instructions and/or data to a programmable processor, including a machine-readable medium that receives machine instructions as a machine-readable signal. The term "machine-readable signal" refers to any signal used to provide machine instructions and/or data to a programmable processor.

[00290] Implementations of the subject matter and the functional operations described in this specification can be implemented in digital electronic circuitry, or in computer software, firmware, or hardware, including the structures disclosed in this specification and their structural equivalents, or in combinations of one or more of them. Embodiments of the subject matter described in this specification can be implemented as one or more computer program products, i.e., one or more modules of computer program instructions encoded on a computer readable medium for execution by, or to control the operation of, data processing apparatus. The computer readable medium can be a machine-readable storage device, a machine-readable storage substrate, a memory device, a composition of matter effecting a machine-readable propagated signal, or a combination of one or more of them. The term "data processing apparatus" encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them. A propagated signal is an artificially generated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal, that is generated to encode information for transmission to suitable receiver apparatus.

[00291] A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[00292] The processes and logic flows described in this specification can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

[00293] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Moreover, a computer can be embedded in another device, e.g., a mobile telephone, a personal digital assistant

(PDA), a mobile audio player, a Global Positioning System (GPS) receiver, to name just a few. Computer readable media suitable for storing computer program instructions and data include all forms of non volatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[00294] Implementations of the subject matter described in this specification can be implemented in a computing system that includes a back end component, e.g., as a data server, or that includes a middleware component, e.g., an application server, or that includes a front end component, e.g., a client computer having a graphical user interface or a web browser through which a user can interact with an implementation of the subject matter described is this specification, or any combination of one or more such back end, middleware, or front end components. The components of the system can be interconnected by any form or medium of digital data communication, e.g., a communication network. Examples of communication networks include a local area network ("LAN") and a wide area network ("WAN"), e.g., the Internet.

[00295] The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[00296] While this specification contains many specifics, these should not be construed as limitations on the scope of the invention or of what may be claimed, but rather as descriptions of features specific to particular implementations of the invention. Certain features that are described in this specification in the context of separate implementations can also be implemented in combination in a single implementation. Conversely, various features that are described in the context of a single implementation can also be implemented in multiple implementations separately or in any suitable sub-combination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a sub-combination or variation of a sub-combination.

[00297] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multi-tasking and parallel processing may be advantageous. Moreover, the separation of various system components in the embodiments described above should not be understood as requiring such separation in all embodiments, and it should be understood that the described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

[00298] A number of implementations have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the disclosure. Accordingly, other implementations are within the scope of the following claims. For example, the actions recited in the claims can be performed in a different order and still achieve desirable results.

WHAT IS CLAIMED IS:

1. A robot system (1600) comprising:

a mobile robot (100) having a controller (500) executing a control system (510) for controlling operation of the robot (100);

a cloud computing service (1620) in communication with the controller (500) of the robot (100); and

a remote computing device (310, 1604) in communication with the cloud computing service (1620), the remote computing device (310, 1604) communicating with the robot (100) through the cloud computing service (1620).

2. The robot system (1600) of claim 1, wherein the remote computing device (310, 1604) executes an application (1610, 1610a) for producing a layout map (1700, 1810) of a robot operating environment (10).

3. The robot system (1600) of claim 2, wherein the remote computing device (310, 1604) stores the layout map (1700, 1810) in external cloud storage (1622) using the cloud computing service (1620).

4. The robot system (1600) of claim 3, wherein the controller (500) of the robot (100) accesses the layout map (1700, 1810) through the cloud computing service (1620) for issuing drive commands to a drive system (200) of the robot (100).

5. The robot system (1600) of any of claims 1-4, wherein the remote computing device (310, 1604) executes an application (1610, 1610b) providing remote teleoperation of the robot (100), preferably the application (1610, 1610b) provides controls for at least one of driving the robot (100), altering a pose of the robot (100), viewing video from a camera (320, 450) of the robot (100), and operating a camera (320, 450) of the robot (100).

6. The robot system (1600) of any of claims 1-5, wherein the remote computing device (310, 1604) executes an application (1610, 1610c) providing video conferencing

between a user of the remote computing device (310, 1604) and a third party within view of a camera (320, 450) of the robot (100).

7. The robot system (1600) of any of claims 1-6, wherein the remote computing device (310, 1604) executes an application (1610, 1610d) for scheduling usage of the robot and/or an application (1610, 1610e) for monitoring usage and operation of the robot (100).

8. The robot system (1600) of any of claims 1-7, wherein the remote computing device (310, 1604) comprises a tablet computer.

9. A robot system (1600) comprising:

a mobile robot (100) having a controller (500) executing a control system (510) for controlling operation of the robot (100);

a computing device (310, 1604) in communication with the controller (500);

a cloud computing service (1620) in communication with the computing device (310, 1604); and

a portal (1630) in communication with the cloud computing service (1620.

10. The robot system (1600) of claim 9, wherein the portal (1630) comprises a web-based portal (1630) providing access to content.

11. The robot system (1600) of claim 9 or 10, wherein the portal (1630) receives robot information from the robot (100) through the cloud computing service (1620) and/or the robot (100) receives user information from the portal (1630) through the cloud computing service (1620).

12. The robot system (1600) of any of claims 9-11, wherein the computing device (310) includes a touch screen (312).

13. The robot system (1600) of any of claims 9-12, wherein the computing device (310, 1604) executes an operating system different from an operating system of the controller (500).

14. The robot system (1600) of any of claims 9-13, wherein the computing device (310, 1604) executes at least one application (1610) that collects robot information from the robot (100) and sends the robot information to the cloud computing service (1620).

15. The robot system (1600) of any of claims 9-14, wherein the robot (100) comprises:

a base (120) defining a vertical center axis (Z) and supporting the controller (500);

a holonomic drive system (200) supported by the base (120), the drive system (200) having first, second, and third drive wheels (210a-c), each trilaterally spaced about the vertical center axis (Z) and each having a drive direction (D_R, Drive) perpendicular to a radial axis (X, F, Slip) with respect to the vertical center axis (Z);

an extendable leg (130) extending upward from the base (120); and

a torso (140) supported by the leg (130), actuation of the leg (130) causing a change in elevation (H_T) of the torso (140), the computing device (310) detachably supported above the torso (140).

16. The robot system of claim 15, wherein the robot (100) further comprises:

a neck (150) supported by the torso (140); and

a head (160) supported by the neck (150), the neck (150) capable of panning and tilting the head (160) with respect to the torso (140), the computing device (310) detachably supported by the head (160).

17. A robot system (1600) comprising:

a mobile robot (100) having a controller (500) executing a control system (510) for controlling operation of the robot (100);

a computing device (310) in communication with the controller (500);

a mediating security device (350) controlling communications between the controller (500) and the computing device (310);

a cloud computing service (1620) in communication with the computing device (310); and

a portal (1630) in communication with cloud computing service (1620).

18. The robot system (1600) of claim 17, wherein the mediating security device (350) converts communications between a computing device communication protocol of the computing device (310) and a robot communication protocol of the robot (100).

19. The robot system (1600) of claim 17 or 18, wherein the mediating security device (350) comprises an authorization chip (352) for authorizing communication traffic between the computing device (310) and the robot (100).

20. The robot system (1600) of any of claims 17-19, wherein the computing device (310) communicates wirelessly with the robot controller (500), preferably the computing device (310) comprises a tablet computer and/or is releasably attachable to the robot (100).

21. The robot system (1600) of any of claims 17-20, wherein the portal (1630) comprises a web-based portal (1630) providing access to content.

22. The robot system (1600) of any of claims 17-21, wherein the portal (1630) receives robot information from the robot (100) through the cloud computing service (1620) and/or the robot (100) receives user information from the portal (1630) through the cloud computing service (1620).

23. The robot system (1600) of any of claims 17-22, wherein the computing device (310) accesses cloud storage (1622) using the cloud computing service (1620).

24. The robot system (1600) of any of claims 17-23, wherein the computing device (310) executes at least one application (1610) that collects robot information from the robot (100) and sends the robot information to the cloud computing service (1620).

25. A method of operating a mobile robot (100), the method comprising:

receiving a layout map (1810) corresponding to an environment (10) of the robot (100);

moving the robot (100) in the environment (10) to a layout map location (1812, 1814) on the layout map (1810);

recording a robot map location (1822) on a robot map (1820) corresponding to the environment (10) and produced by the robot (100);

determining a distortion between the robot map (1820) and the layout map (1810) using the recorded robot map locations (1822) and the corresponding layout map locations (1812); and

applying the determined distortion to a target layout map location (1814) to determine a corresponding target robot map location (1824).

26. The method of claim 25, further comprising receiving the layout map (1810) from a cloud computing service (1620).

27. The method of claim 26, further comprising producing the layout map (1810) on an application (1610) executing on a remote computing device (310, 1604) and storing the layout map (1810) on a remote cloud storage device (1622) using the cloud computing service (1620).

28. The method of any of claims 25-27, further comprising:

determining a scaling size, origin mapping, and rotation between the layout map (1810) and the robot map (1820) using existing layout map locations (1812) and recorded robot map locations (1822); and

resolving a target robot map location (1824) corresponding to the target layout map location (1814).

29. The method of claim 28, further comprising applying an affine transformation to the determined scaling size, origin mapping, and rotation to resolve the target robot map location (1824).

30. The method of any of claims 25-29, further comprising determining a triangulation (1910) between layout map locations (1812, 1912) that bound the target layout map location (1814, 1914).

31. The method of claim 30, further comprising determining a scale, rotation, translation, and skew between a triangle (1910) mapped in the layout map (1810) and a corresponding triangle (1920) mapped in the robot map (1820) and applying the determined scale, rotation, translation, and skew to the target layout map location (1814, 1914) to determine the corresponding target robot map location (1824, 1924).

32. The method of any of claims 25-31, further comprising:

determining distances between all layout map locations (1812, 1912) and the target layout map location (1814, 1914);

determining a centroid (2012) of the layout map locations (1812, 1912);

determining a centroid (2022) of all recorded robot map locations (1822, 1922); and

for each layout map location (1812, 1912), determining a rotation and a length scaling to transform a first vector (2014) running from the layout map centroid (2012) to the target layout location (1814, 1914) into a second vector (2024) running from the robot map centroid (2022) to the target robot map location (1824, 1924).

33. The method of any of claims 25-32, further comprising producing the robot map (1810) using a sensor system (400) of the robot (100).

34. The method of claim 33, further comprising

emitting light onto a scene (10) of the environment;

receiving reflections of the emitted light off surfaces of the scene (10);
determining a distance of each reflecting surface; and
constructing a three-dimensional depth map (1700) of the scene (10).

35. The method of claim 34, further comprising emitting a speckle pattern of light onto the scene (10) and receiving reflections of the speckle pattern from the scene (10), preferably storing reference images of the speckle pattern as reflected off a reference object (12) in the scene (10), the reference images captured at different distances (Z_n) from the reference object (12).

36. The method of claim 35, further comprising capturing at least one target image of the speckle pattern as reflected off a target object (12) in the scene (10) and comparing the at least one target image with the reference images for determining a distance of the reflecting surfaces of the target object (12).

37. The method of claim 36, further comprising determining a primary speckle pattern on the target object (12) and computing at least one of a respective cross-correlation and a decorrelation between the primary speckle pattern and the speckle patterns of the reference images.

38. The method of claim 36 or 37, further comprising maneuvering the robot (100) with respect to the target object (12) based on the determined distances of the reflecting surfaces of the target object (12).

39. The method of claim 34, further comprising determining a time-of-flight between emitting the light and receiving the reflected light and determining a distance to the reflecting surfaces of the scene (10).

40. The method of any of claims 34-39, further comprising emitting the light onto the scene (10) in intermittent pulses, preferably altering a frequency of the emitted light pulses.

41. A robot system (1600) comprising:

a mobile robot (100) comprising:

a controller (500) executing a control system (510) for controlling operation of the robot (100); and

a sensor system (400) in communication with the controller (500); and

a cloud computing service (1620) in communication with the controller (500) of the robot (100);

wherein the cloud computing service (1620):

receives data (1601) from the controller (500);

processes the data (1601); and

returns processed resultant (1607, 1609) to the controller (500).

42. The robot system (1600) of claim 41, wherein the cloud computing service (1620) at least temporarily stores the received data (1601, 1603) in cloud storage (1622) and optionally discards the stored data (1601, 1603) after processing the data (1601, 1603).

43. The robot system (1600) of claim 41 or 42, wherein the robot (100) comprises:

a camera (320) in communication with the controller (500) and capable of obtaining images of a scene (10) about the robot (100); and/or

a volumetric point cloud imaging device (450) in communication with the controller (500) and capable of obtaining a point cloud from a volume of space about the robot (100), preferably a volume of space that includes a floor plane (5) in a direction of movement (F) of the robot (100);

wherein the controller (500) communicates image data (1601) to the cloud computing service (1620).

44. The robot system (1600) of any of claims 41-43, wherein the data (1601) comprises raw sensor data and/or data having associated information from the sensor system (400), preferably the data (1601) comprises image data having at least one of accelerometer data traces, odometry data, and a timestamp.

45. The robot system (1600) of any of claims 41-44, wherein the cloud computing service (1620) receives image data (1601) from the controller (500) of a scene (10) about the robot (100) and processes the image data (1601) into a 3-D map (1605) and/or a model (1609) of the scene (10).

46. The robot system (1600) of claim 45, wherein the cloud computing service (1620) provides a 2-D height map (1607) and/or the model (1609) to the controller (500), the cloud computing service (1620) computing the 2-D height map (1607) from the 3-D map (1605).

47. The robot system (1600) of claim 45 or 46, wherein the cloud computing service (1620) receives the image data (1601) periodically and processes the received image data (1601) after accumulating a threshold image data set (1603).

48. The robot system (1600) of any of claims 41-47, wherein the controller (500) communicates the data (1601) to the cloud computing service (1620) wirelessly through a portable computing device (310) in communication with the controller (500), and preferably removably attachable to the robot (100).

49. The robot system (1600) of any of claims 41-48, wherein the controller (500) buffers the data (1601) and sends the data (1601) to the cloud computing service (1620) periodically.

50. The robot system (1600) of any of claims 41-49, wherein the sensor system (400) comprises at least one of a camera (320), a 3-D imaging sensor (450), a sonar sensor, an ultrasonic sensor, LIDAR, LADAR, an optical sensor, and an infrared sensor.

51. A method of operating a mobile robot (100), the method comprising:
maneuvering the robot (100) about a scene (10);
receiving sensor data (1601) indicative of the scene (10);

communicating the sensor data (1601) to a cloud computing service (1620) that processes the received sensor data (1601) and communicates a process resultant (1607, 1609) to the robot (100); and

maneuvering the robot (100) in the scene (10) based on the received process resultant (1607, 1609).

52. The method of claim 51, further comprising:

emitting light onto the scene (10) about the robot (100); and

capturing images of the scene (10) along a drive direction (F) of the robot (100), the images comprising at least one of (a) a three-dimensional depth image, (b) an active illumination image, and (c) an ambient illumination image;

wherein the sensor data (1601) comprises the images and the process resultant (1607, 1609) comprises a map (1607) or a model (1609) of the scene (10).

53. The method of claim 52, further comprising:

emitting a speckle pattern of light onto the scene (10);

receiving reflections of the speckle pattern from an object (12) in the scene (10);

storing reference images in cloud storage (1622) of the cloud computing service (1620) of the speckle pattern as reflected off a reference object (12) in the scene (10), the reference images captured at different distances (Z_n) from the reference object (12); and

capturing at least one target image of the speckle pattern as reflected off a target object (12) in the scene (10) and communicating the at least one target image to the cloud computing service (1620);

wherein the cloud computing service (1620) compares the at least one target image with the reference images for determining a distance (ΔZ) of the reflecting surfaces of the target object (12).

54. The method of claim 53, further comprising determining a primary speckle pattern on the target object (12) and computing at least one of a respective cross-correlation and a decorrelation between the primary speckle pattern and the speckle patterns of the reference images.

55. The method of any of claims 51-54, wherein the cloud computing service (1620) at least temporarily stores the received sensor data (1601) in cloud storage (1622) and optionally discards the stored sensor data (1601) after processing the data (1601).

56. The method of any of claims 51-55, wherein the sensor data (1601) comprises image data having associated sensor system data, preferably the sensor system data comprising at least one of accelerometer data traces, odometry data, and a timestamp.

57. The method of any of claims 51-56, wherein the cloud computing service (1620) receives image data (1601) from the robot (100) and processes the image data (1601) into a 3-D map (1605) and/or model (1609) of the scene (10).

58. The method of claim 57, wherein the cloud computing service (1620) provides a 2-D height map (1607) and/or the model (1609) to the robot (100), the cloud computing service (1620) computing the 2-D height map (1607) from the 3-D map (1605).

59. The method of any of claims 51-56, further comprising periodically communicating the sensor data (1601) to the cloud computing service (1620), the cloud computing service (1620) processing the received image data (1601) after accumulating a threshold sensor data set (1603).

60. The method of any of claims 51-59, further comprising communicating the sensor data (1601) to the cloud computing service (1620) wirelessly through a portable computing device (310) in communication with the robot (100), and preferably removably attachable to the robot (100).

61. A method of navigating a mobile robot (100), the method comprising:

capturing a streaming sequence (1615) of dense images (1611) of a scene (10) about the robot (100) along a locus of motion of the robot (100) at a real-time capture rate;

associating annotations (1613) with at least some of the dense images (1611);

sending the dense images (1611) and annotations (1613) to a remote server (1620) at a send rate, the send rate being slower than the real-time capture rate;

receiving a data set (1607, 1617) from the remote server (1620) after a processing time interval, the data set (1607, 1617) derived from and representing at least a portion of the sequence (1615) of dense images (1611) and corresponding annotations (1613), the data set (1607, 1617) excluding raw image data of the sequence (1615) of dense images (1611);

moving the robot (100) with respect to the scene (10) based on the received data set (1607, 1617).

62. The method of claim 61, further comprising sending the dense images (1611) and annotations (1613) to a local server and buffer (1640) and then sending the dense images (1611) and annotations (1613) to the remote server (1620) at a send rate slower than the real-time capture rate.

63. The method of claim 61 or 62, wherein the annotations (1613) comprise:

a time stamp, preferably of absolute time reference corresponding to at least some of the dense images (1611);

and pose-related sensor data, preferably the pose-related sensor data includes at least one of odometry data, accelerometer data, tilt data, and angular rate data.

64. The method of any of claims 61-63, wherein associating annotations (1613) comprises associating annotations (1613) that reflect hazard events with dense images (1611) captured in a time interval relative to a hazard response of the robot (100).

65. The method of any of claims 61-64, wherein associating annotations (1613) comprises associating key-frame identifiers with a subset of the dense images (1611).

66. The method of any of claims 61-65, wherein the annotations (1613) comprise a sparse set of 3-D points derived from structure and motion recovery of features tracked between dense images (1611) of the streaming sequence (1615) of dense images (1611).

67. The method of claim 66, wherein the annotations (1613) comprise camera parameters, preferably the camera parameters include a camera pose relative to individual 3-D points of the sparse set of 3-D points.

68. The method of any of claims 61-67, wherein the annotations (1613) comprise labels of traversable and non-traversable regions of the scene (10).

69. The method of any of claims 61-68, wherein the data set (1607, 1617) comprises one or more texture maps (1607) extracted from the dense images (1611) and/or a terrain map (1607) representing features within the dense images (1611) of the scene (10).

70. The method of any of claims 61-69, wherein the data set (1607, 1617) comprises a trained classifier (1625) for classifying features within new dense images (1611) captured of the scene (10).

71. A method of abstracting mobile robot environmental data, the method comprising:

receiving a sequence (1615) of dense images (1611) of a robot environment (10) from a mobile robot (100) at a receiving rate, the dense images (1611) captured along a locus of motion of the mobile robot (100) at a real-time capture rate, the receiving rate being slower than the real-time capture rate;

receiving annotations (1613) associated with at least some of the dense images (1611) in the sequence (1615) of dense images (1611);

dispatching a batch processing task for reducing dense data within least some of the dense images (1611) to a data set (1607, 1617) representing at least a portion of the sequence (1615) of dense images (1611);

transmitting the data set (1617) to the mobile robot (100), the data set (1607, 1617) excluding raw image data of the sequence (1615) of dense images (1611).

72. The method of claim 71, wherein the batch processing task comprises:

processing the sequence (1615) of dense images (1611) into a dense 3-D model (1609) of the robot environment (10); and

processing the dense 3-D model (1609) into a terrain model (1607) for a coordinate system of 2-D location and at least one height from a floor plane (G).

73. The method of claim 72, wherein the terrain model (1607) is for a coordinate system of 2-D location and a plurality of occupied and unoccupied height boundaries from a floor plane (G).

74. The method of any of claims 71-73, wherein the batch processing task comprises accumulating dense image sequences (1615, 1615a) corresponding to a plurality of robot environments (10).

75. The method of any of claims 71-74, wherein the batch processing task comprises a plurality of classifiers (1625) and/or training one or more classifiers (1625) on the sequence of dense images (1611).

76. The method of any of claims 71-75, wherein the batch processing task comprises:

associating annotations (1613) that reflect hazard events with dense images (1611) captured in a time interval relative to a hazard response of the mobile robot (100); and

training a classifier (1625) of hazard-related dense images (1611) using the associated hazard event annotations (1613) and corresponding dense images (1611) as training data, preferably to provide a data set (1603, 1607, 1617) of model parameters for the classifier (1625).

77. The method of claim 76, wherein the classifier (1625) includes at least one Support Vector Machine that constructs at least one hyperplane for classification, and the model parameters define a trained hyperplane capable of classifying a data set (1603, 1607, 1617) into hazard-related classifications.

78. The method of claim 77, wherein the model parameters include sufficient parameters to define a kernel of the Support Vector Machine and a soft margin parameter.

79. The method of any of claims 71-78, wherein the batch processing task comprises instantiating a scalable plurality of virtual processes (1621) proportionate to a scale of the dense image sequence (1615, 1615a) to be processed, at least some of the virtual processes (1621) being released after transmission of the data set (1607, 1617).

80. The method of any of claims 71-79, wherein the batch processing task comprises instantiating a scalable plurality of virtual storage (1622) proportionate to a scale of the dense image sequence (1615, 1615a) to be stored, at least some of the virtual storage (1622) being released after transmission of the data set (1607, 1617).

81. The method of any of claims 71-80, wherein the batch processing task comprises distributing a scalable plurality of virtual servers (1621) according to one of geographic proximity to the mobile robot (100) and/or network traffic from a plurality of mobile robots (100).

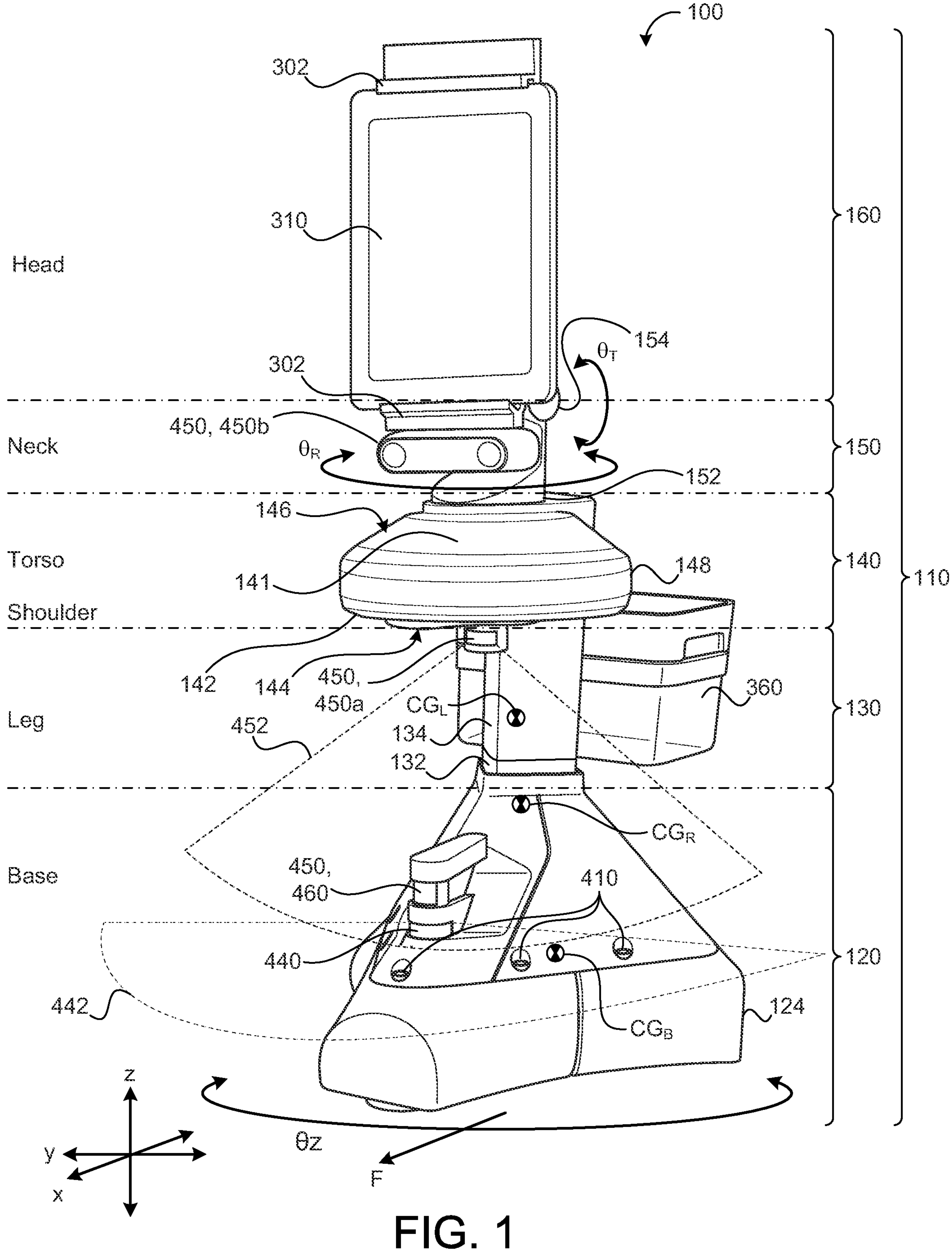
100
302
310
Head
154
θ_T
302
160
Neck
450, 450b
θ_R
150
152
146
Torso
141
148
140
Shoulder
110
142
144
450,
450a
360
Leg
CG_L
452
134
132
130
CG_R
Base
450,
460
410
440
120
442
124
CG_B
z
y
x
θz
F

FIG. 1

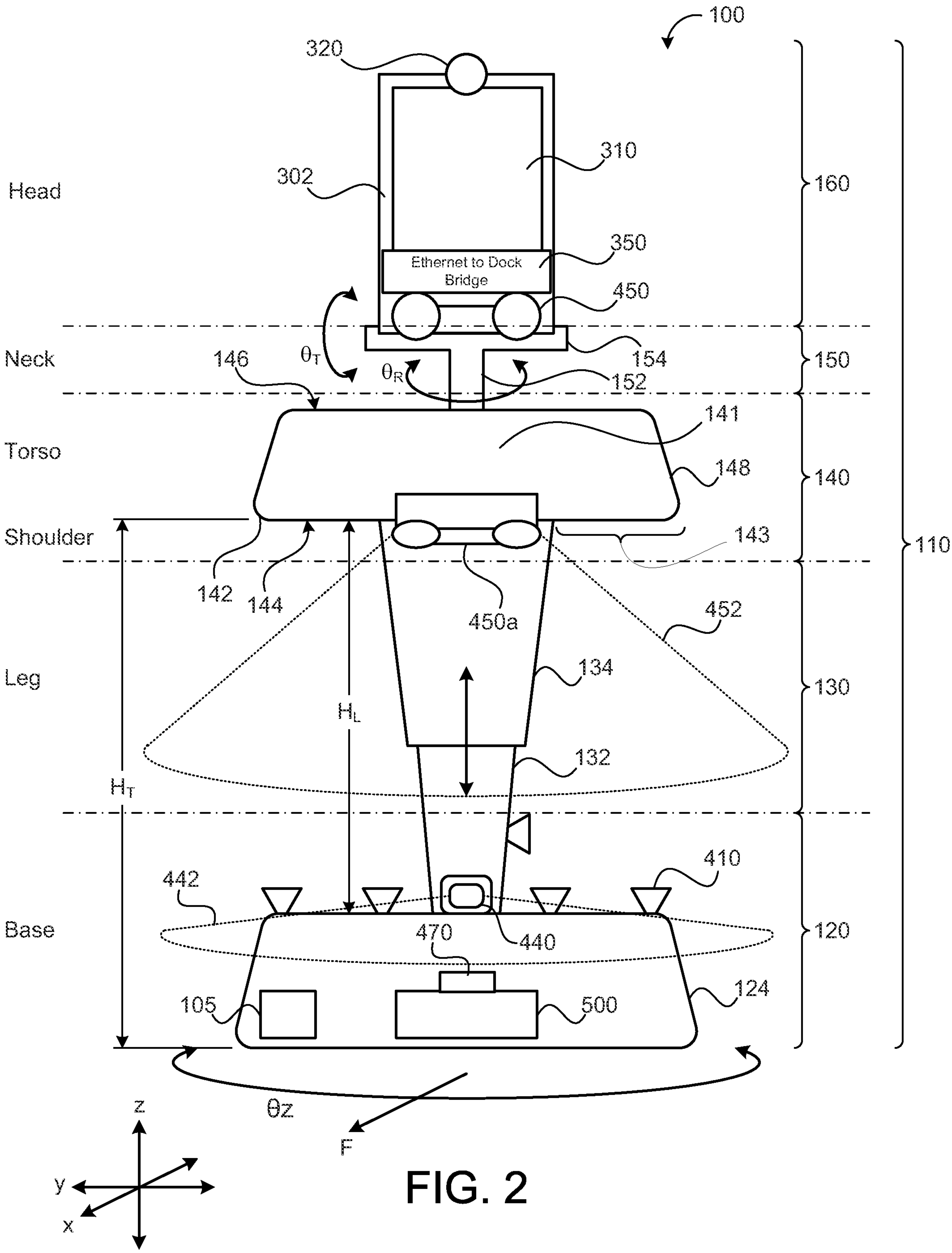
100
320
310
302
Head
160
350
Ethernet to Dock Bridge
450
Neck
146
θ_T
θ_R
154
150
152
141
Torso
148
140
Shoulder
143
110
142
144
450a
452
Leg
134
130
H_L
132
H_T
410
442
Base
470
440
120
105
500
124
θz
F
z
y
x

FIG. 2

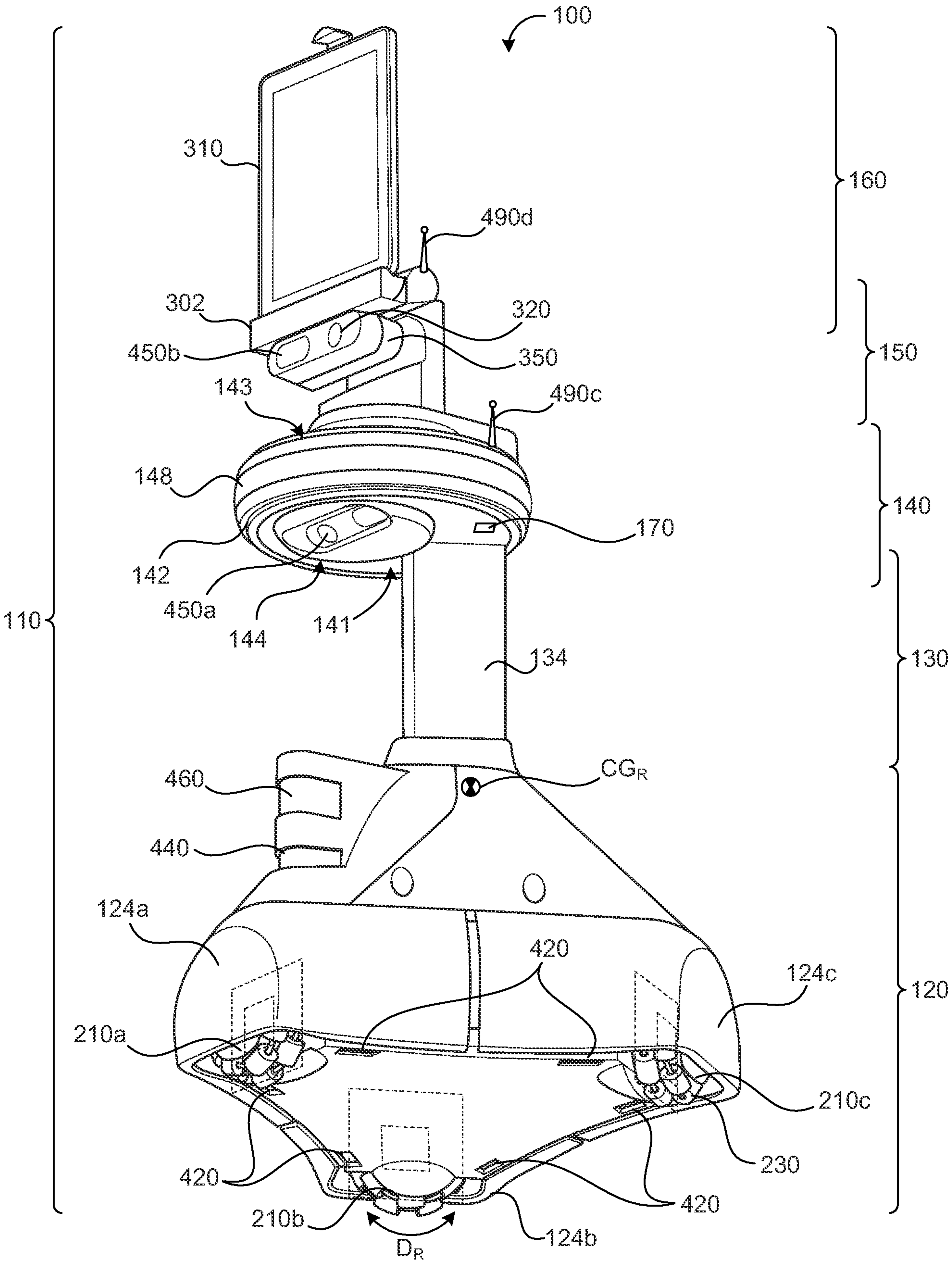
100
310
160
490d
302
320
450b
350
150
143
490c
148
140
170
142
450a
144
141
110
134
130
460
CG_R
440
124a
420
124c
120
210a
210c
420
230
420
210b
124b
D_R

FIG. 3

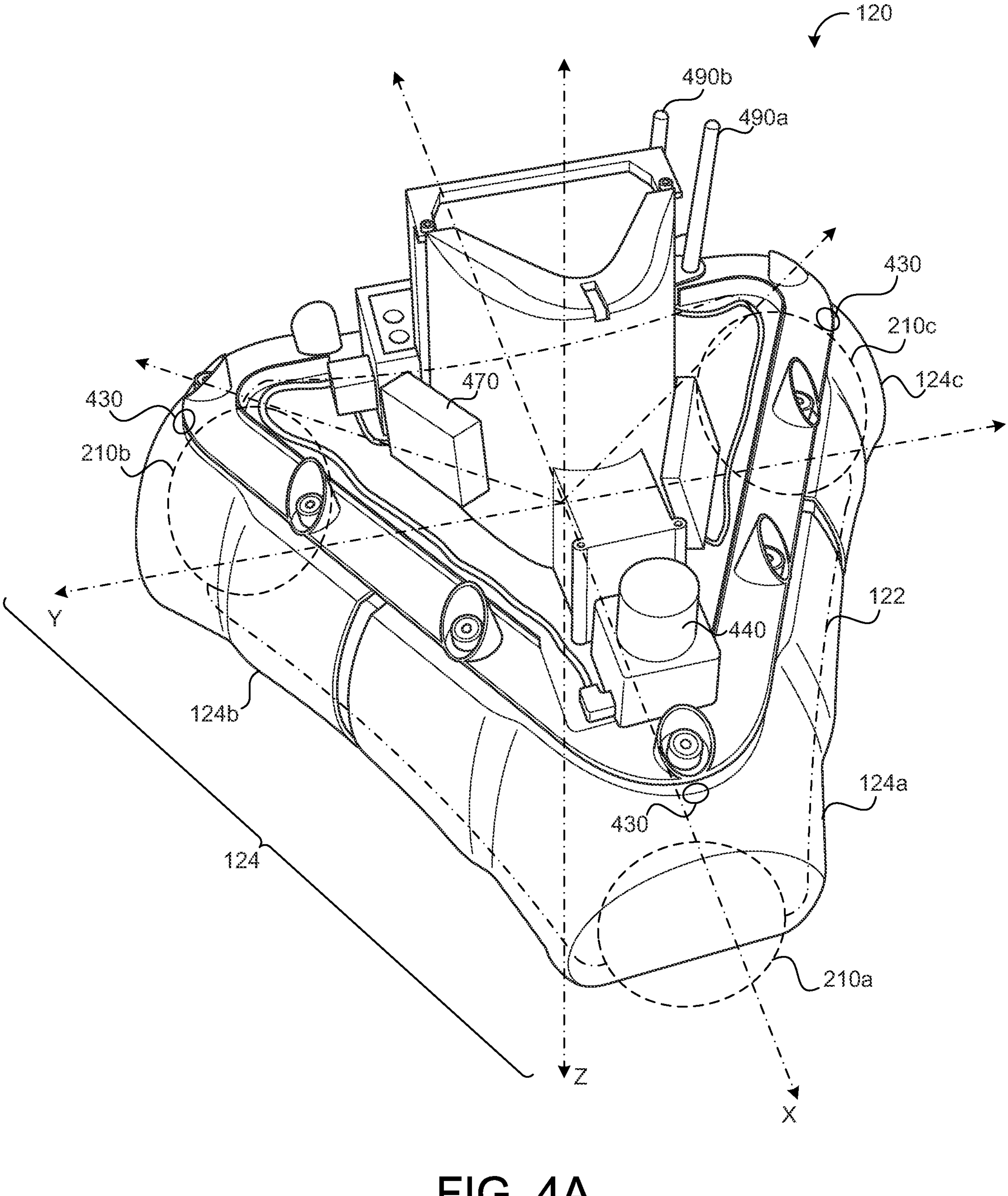
120
490b
490a
430
210c
124c
470
430
210b
122
440
Y
124b
124a
430
124
210a
Z
X

FIG. 4A

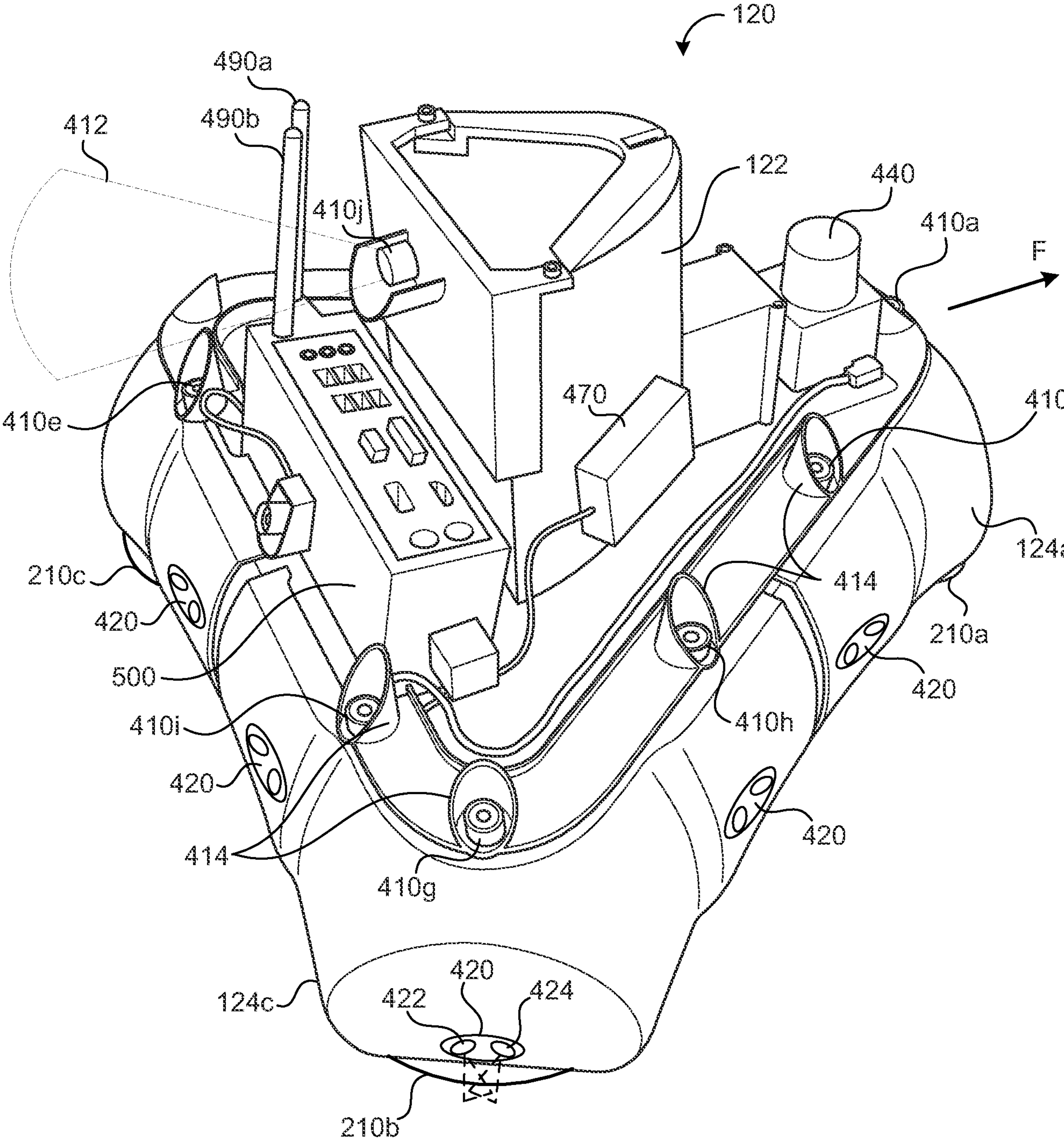
120
490a
490b
412
410j
122
440
410a
F
410e
470
410c
124a
414
210c
420
210a
420
500
410i
410h
420
420
414
410g
124c
420
422
424
210b

FIG. 4B

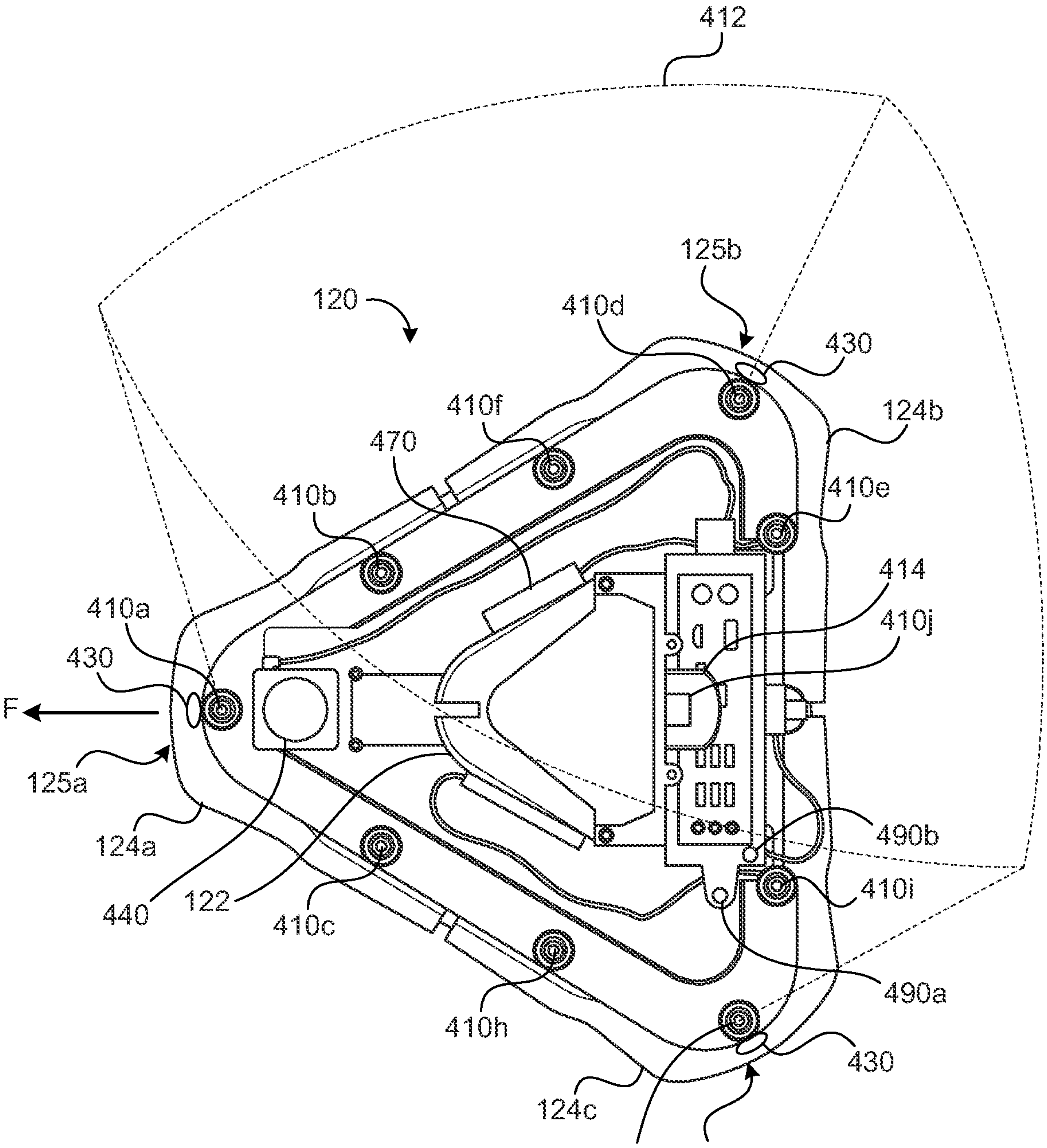
412
125b
120
410d
430
410f
124b
470
410e
410b
414
410a
410j
430
F
125a
124a
490b
410i
440
122
410c
490a
410h
430
124c
410g
125c

FIG. 4C

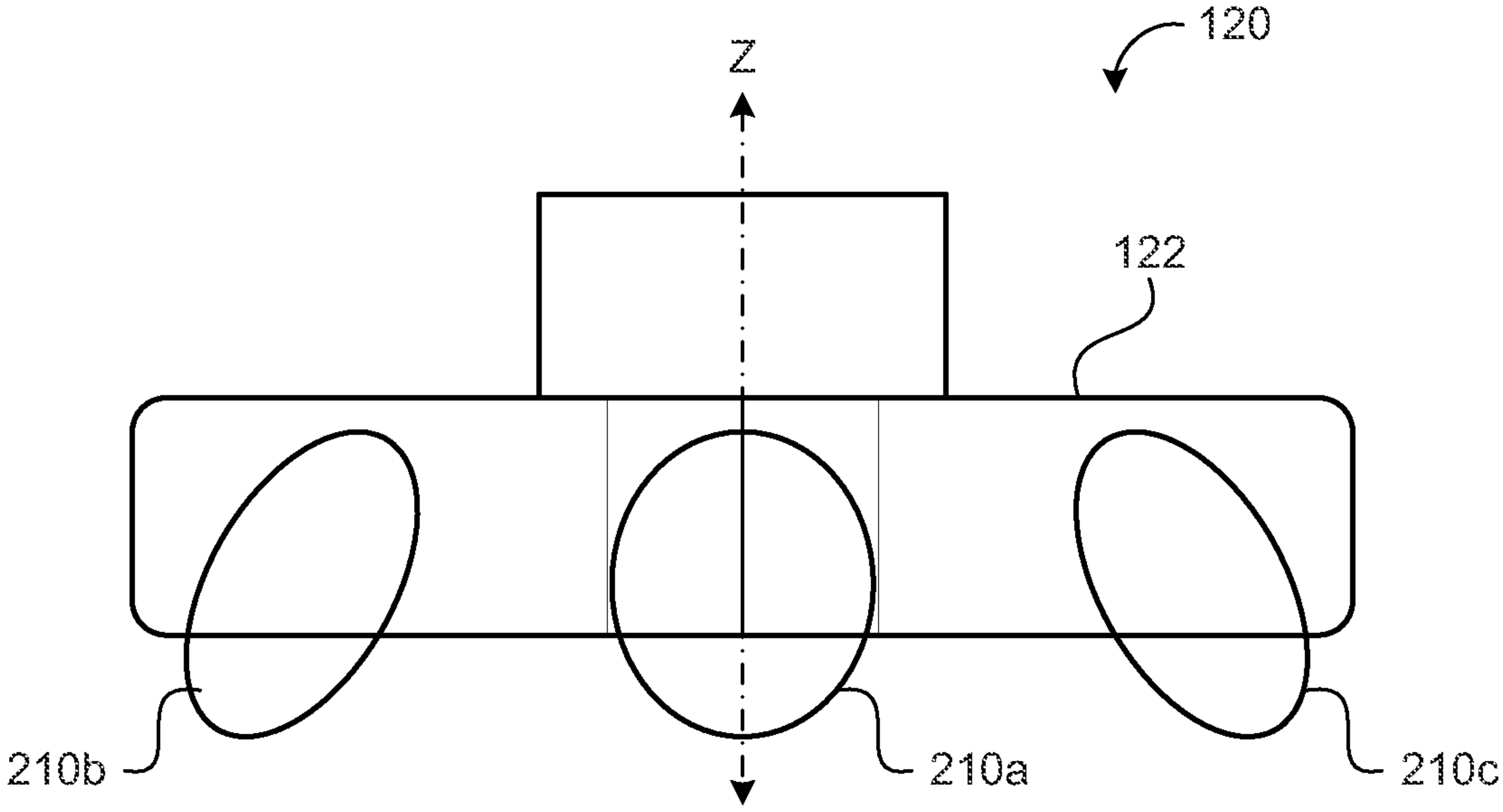
120
Z
122
210b
210a
210c

FIG. 5A

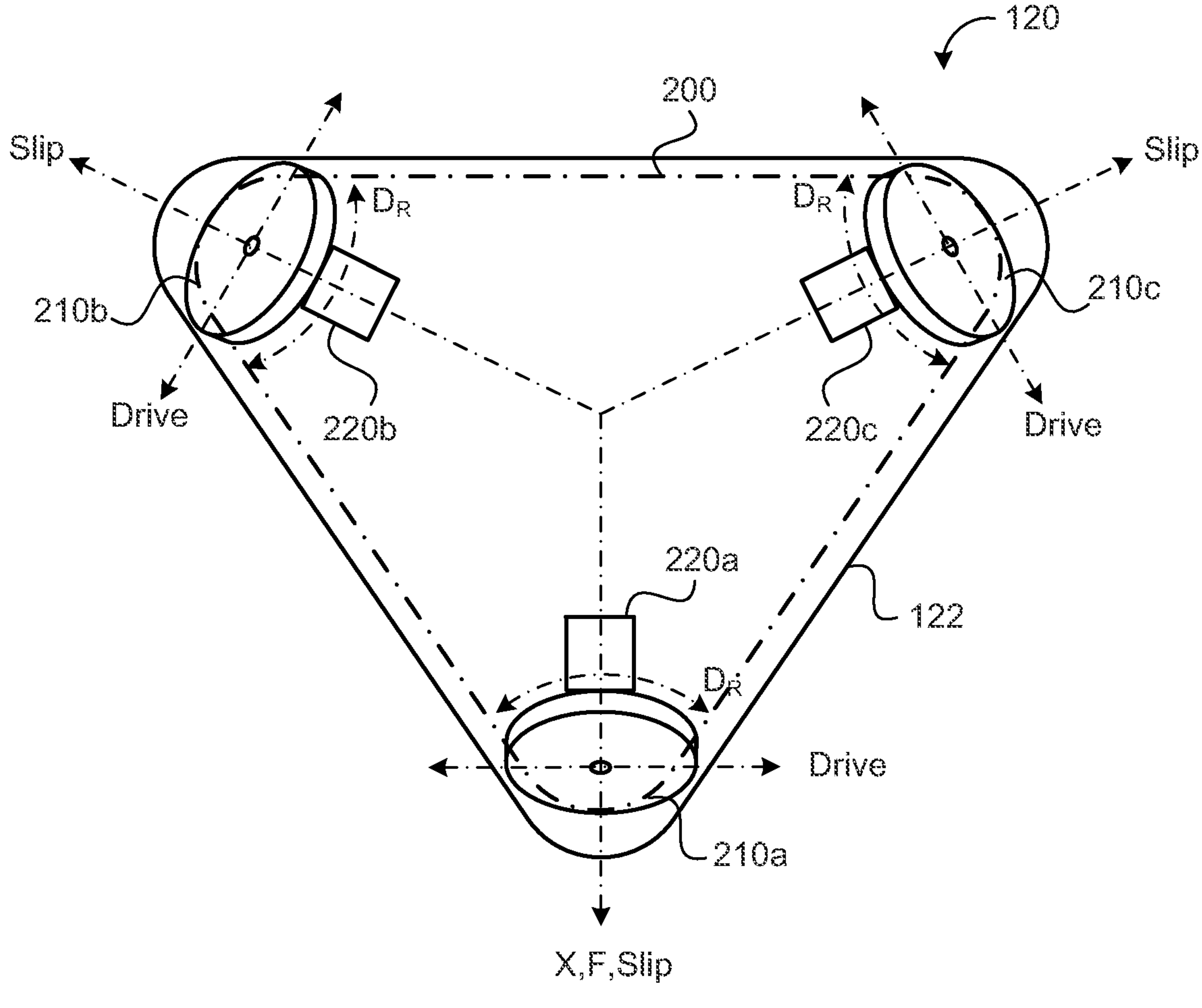
120
200
Slip
Slip
D_R
D_R
210b
210c
Drive
220b
220c
Drive
220a
122
D_R
Drive
210a
X,F,Slip

FIG. 5B

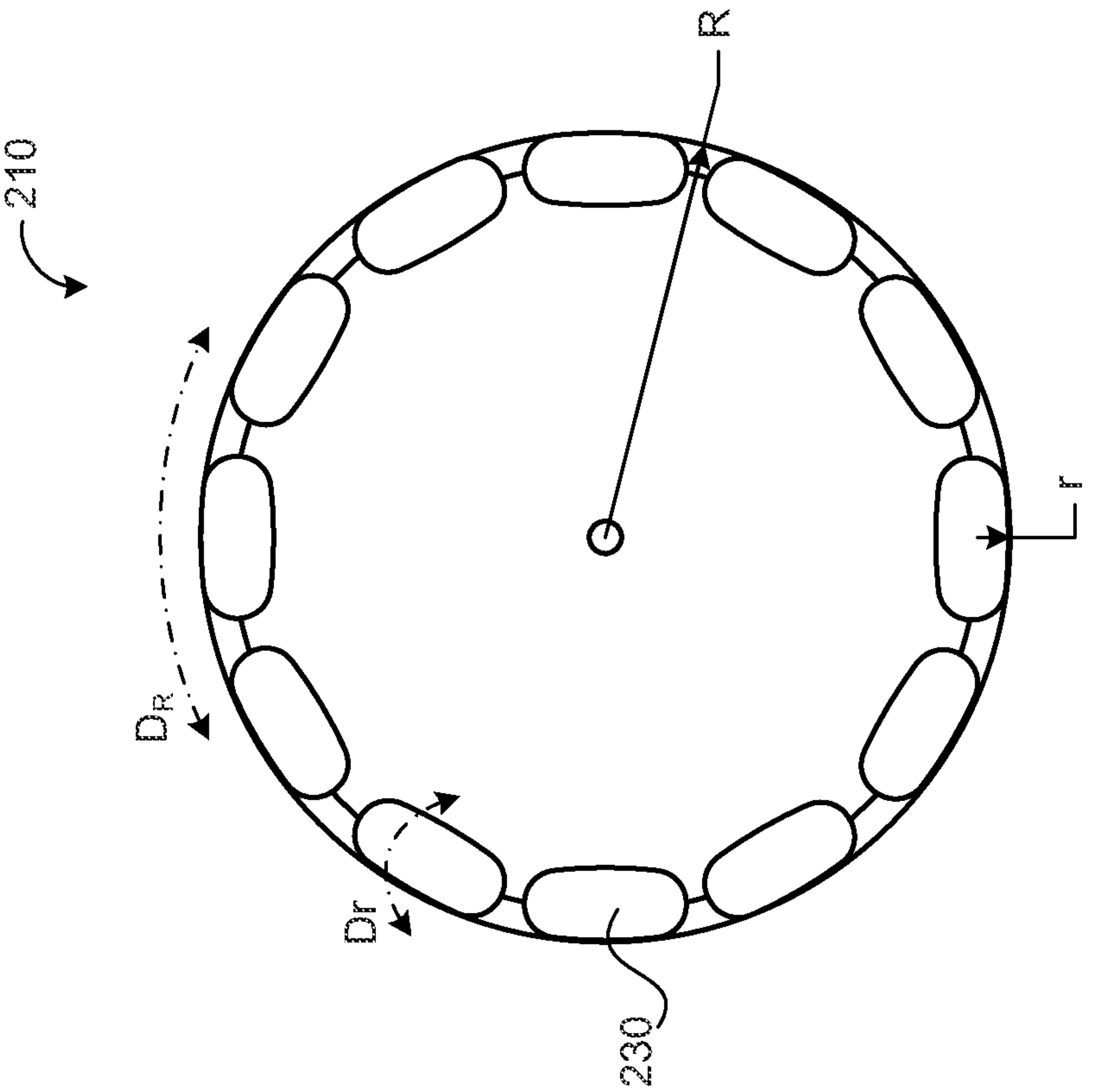

FIG. 5D

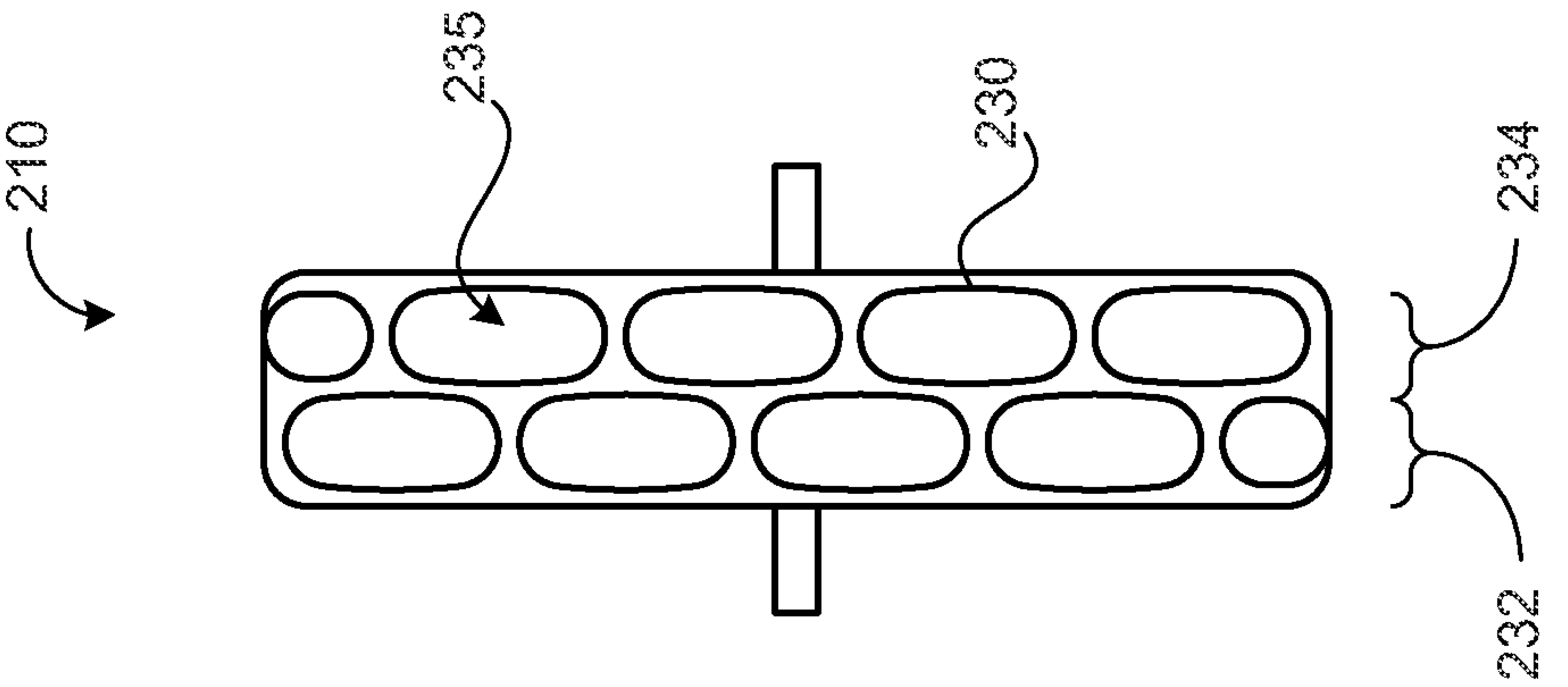

FIG. 5C

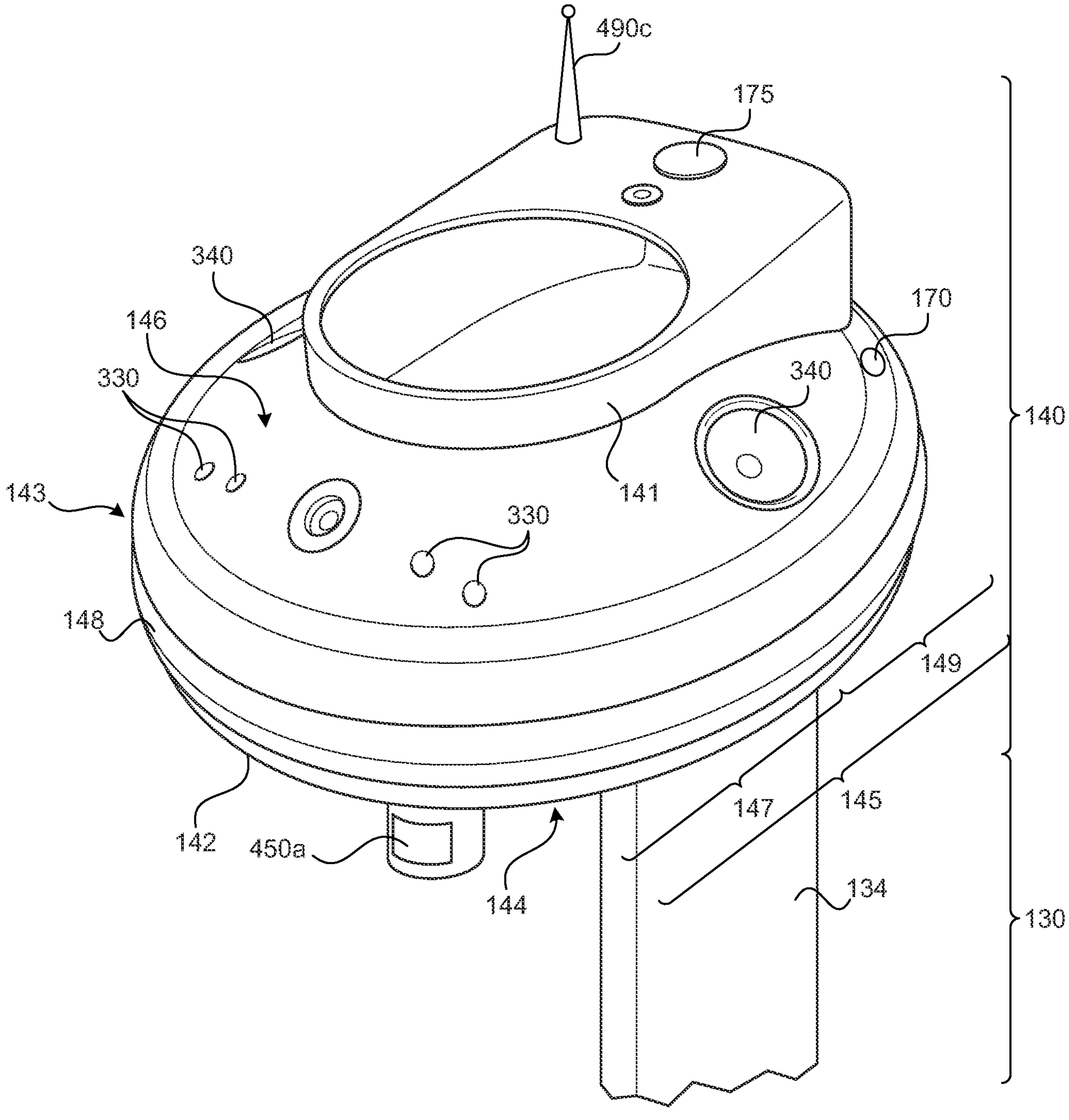
490c
175
340
146
170
340
330
140
143
141
330
148
149
147
145
142
450a
144
134
130

FIG. 6A

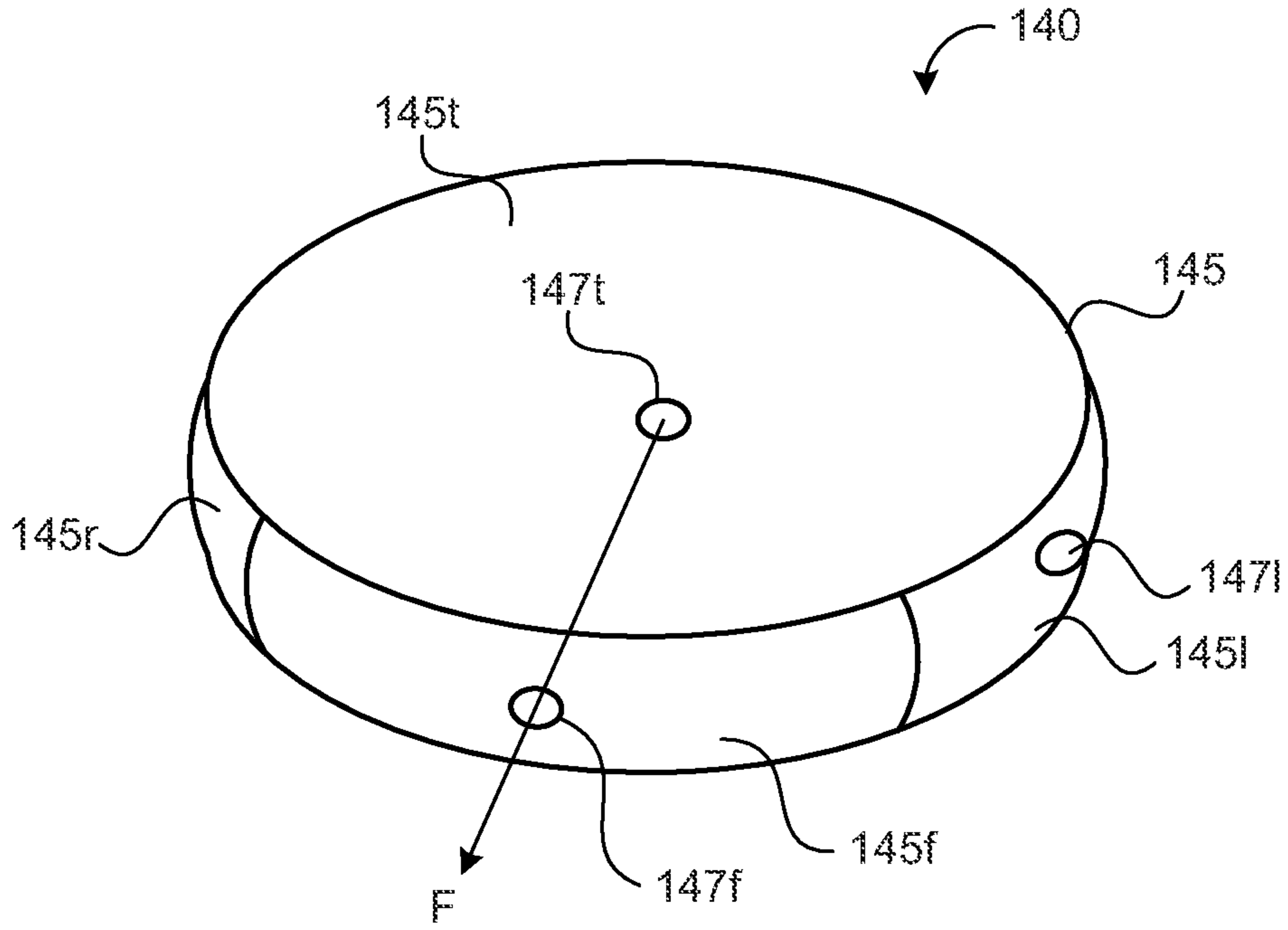
140
145t
147t
145
145r
147l
145l
147f
145f
F

FIG. 6B

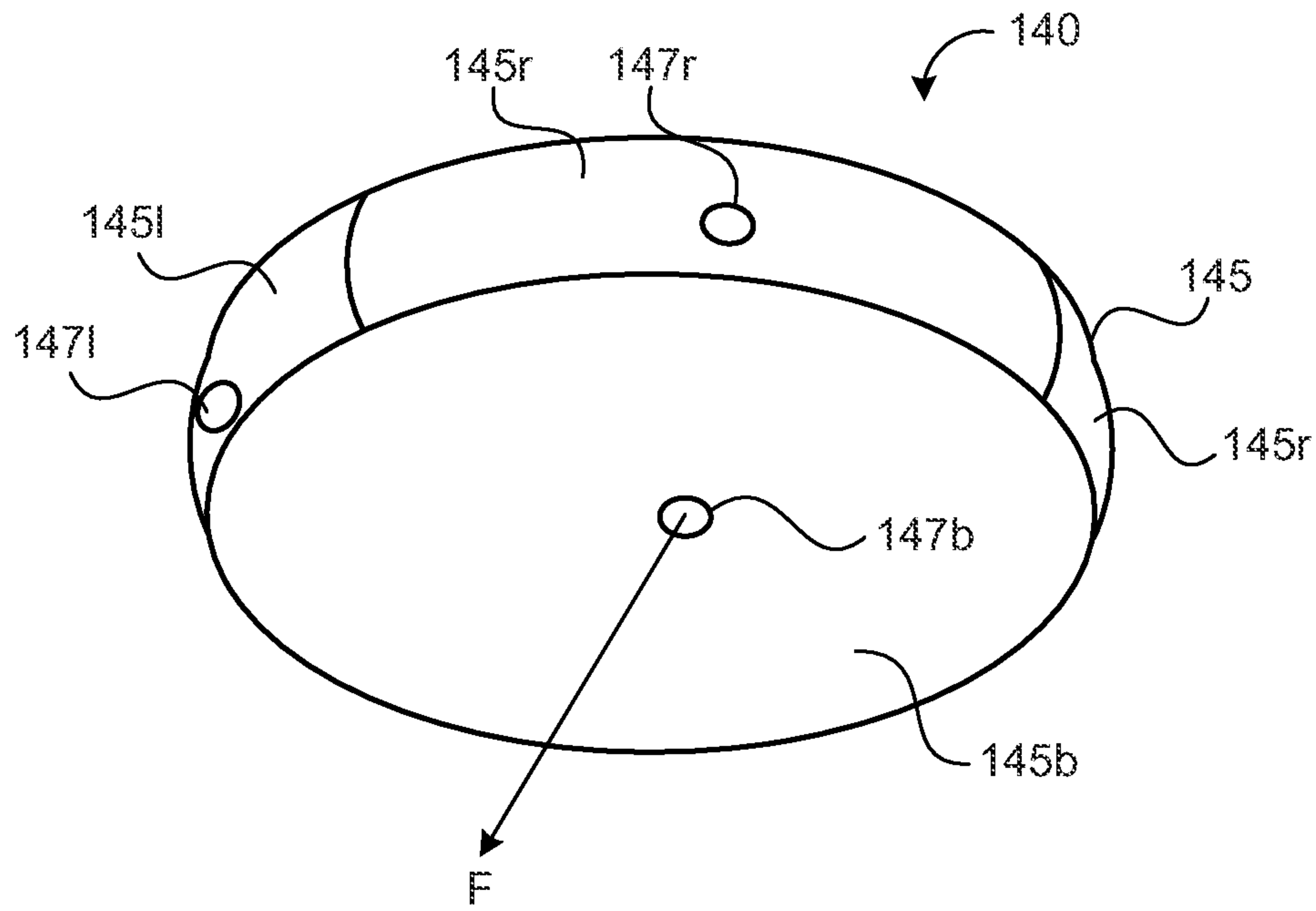
140
145r
147r
145l
145
147l
145r
147b
145b
F

FIG. 6C

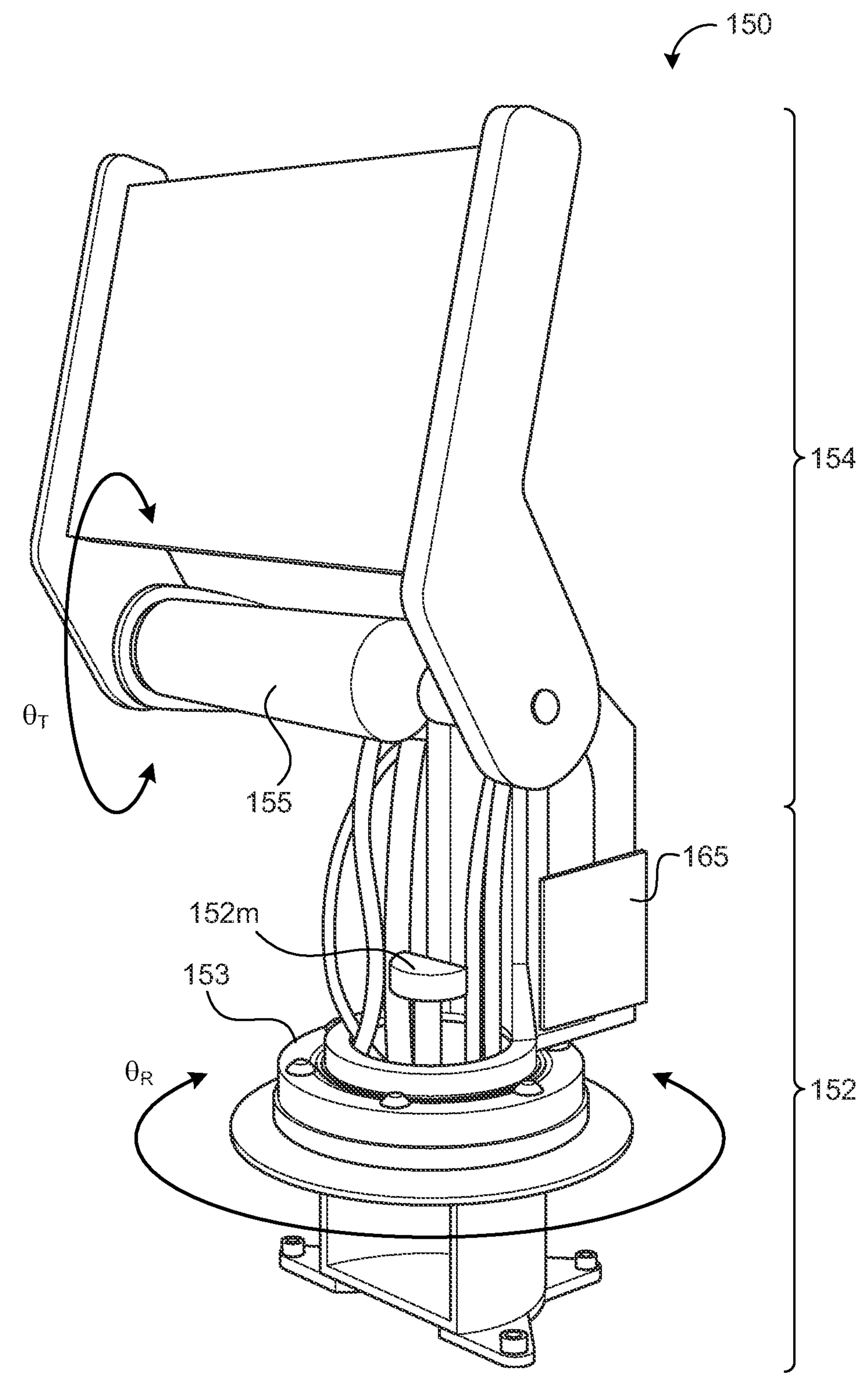

FIG. 7

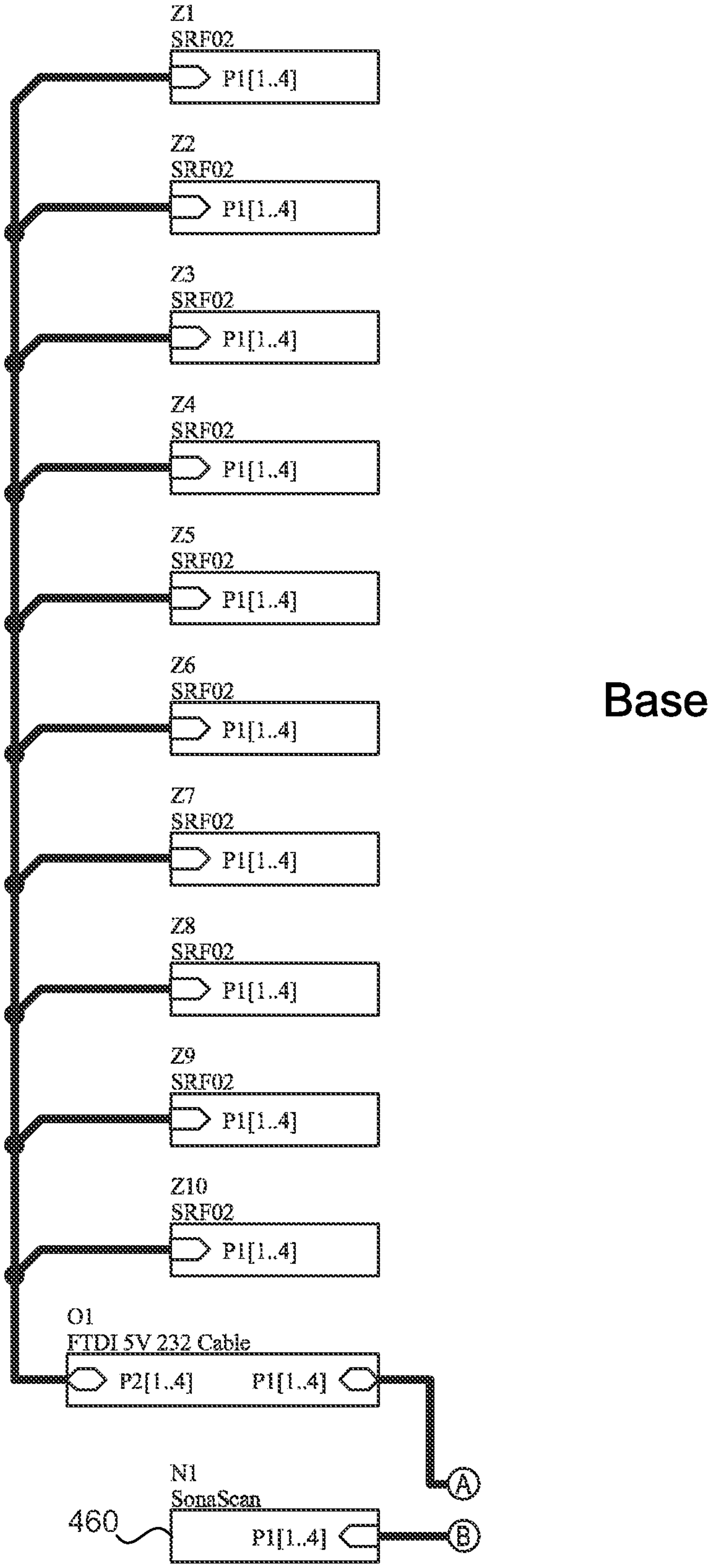
Z1
SRF02
P1[1..4]
Z2
SRF02
P1[1..4]
Z3
SRF02
P1[1..4]
Z4
SRF02
P1[1..4]
Z5
SRF02
P1[1..4]
Z6
SRF02
P1[1..4]
Z7
SRF02
P1[1..4]
Z8
SRF02
P1[1..4]
Z9
SRF02
P1[1..4]
Z10
SRF02
P1[1..4]
Base
O1
FTDI 5V 232 Cable
P2[1..4]
P1[1..4]
A
N1
SonaScan
460
P1[1..4]
B

FIG. 8A

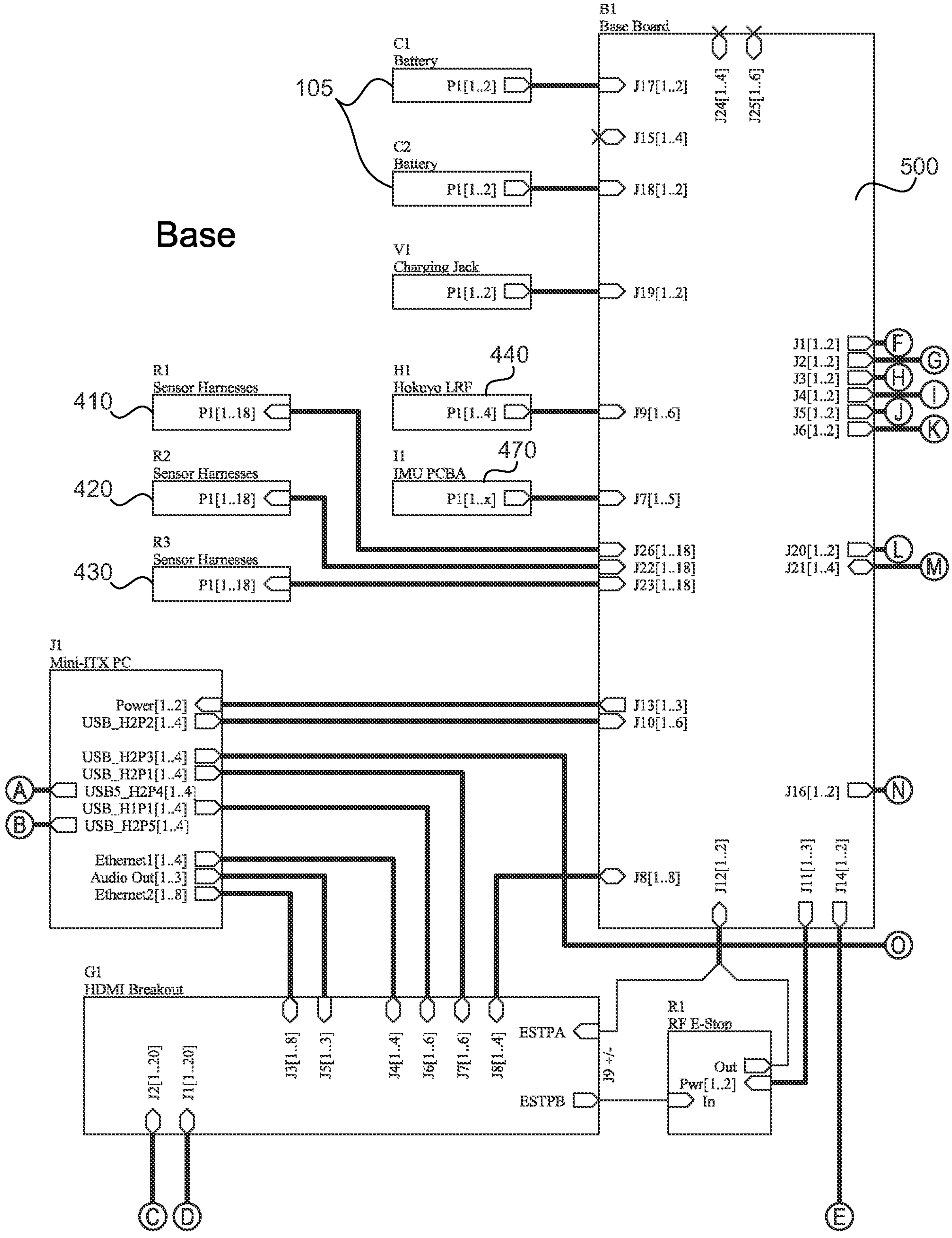
B1
Base Board
C1
Battery
P1[1..2]
J17[1..2]
J24[1..4]
J25[1..6]
105
J15[1..4]
C2
Battery
P1[1..2]
J18[1..2]
500
Base
V1
Charging Jack
P1[1..2]
J19[1..2]
J1[1..2]
J2[1..2]
J3[1..2]
J4[1..2]
J5[1..2]
J6[1..2]
440
R1
Sensor Harnesses
H1
Hokuyo LRF
410
P1[1..18]
P1[1..4]
J9[1..6]
R2
Sensor Harnesses
470
I1
IMU PCBA
420
P1[1..18]
P1[1..x]
J7[1..5]
R3
Sensor Harnesses
J26[1..18]
J20[1..2]
430
P1[1..18]
J22[1..18]
J21[1..4]
J23[1..18]
J1
Mini-ITX PC
Power[1..2]
J13[1..3]
USB_H2P2[1..4]
J10[1..6]
USB_H2P3[1..4]
USB_H2P1[1..4]
USB5_H2P4[1..4]
J16[1..2]
USB_H1P1[1..4]
USB_H2P5[1..4]
J12[1..2]
J11[1..3]
J14[1..2]
Ethernet1[1..4]
Audio Out[1..3]
J8[1..8]
Ethernet2[1..8]
G1
HDMI Breakout
R1
RF E-Stop
ESTPA
J3[1..8]
J5[1..3]
J4[1..4]
J6[1..6]
J7[1..6]
J8[1..4]
J9 +/-
Out
Pwr[1..2]
J2[1..20]
J1[1..20]
ESTPB
In
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O

FIG. 8B

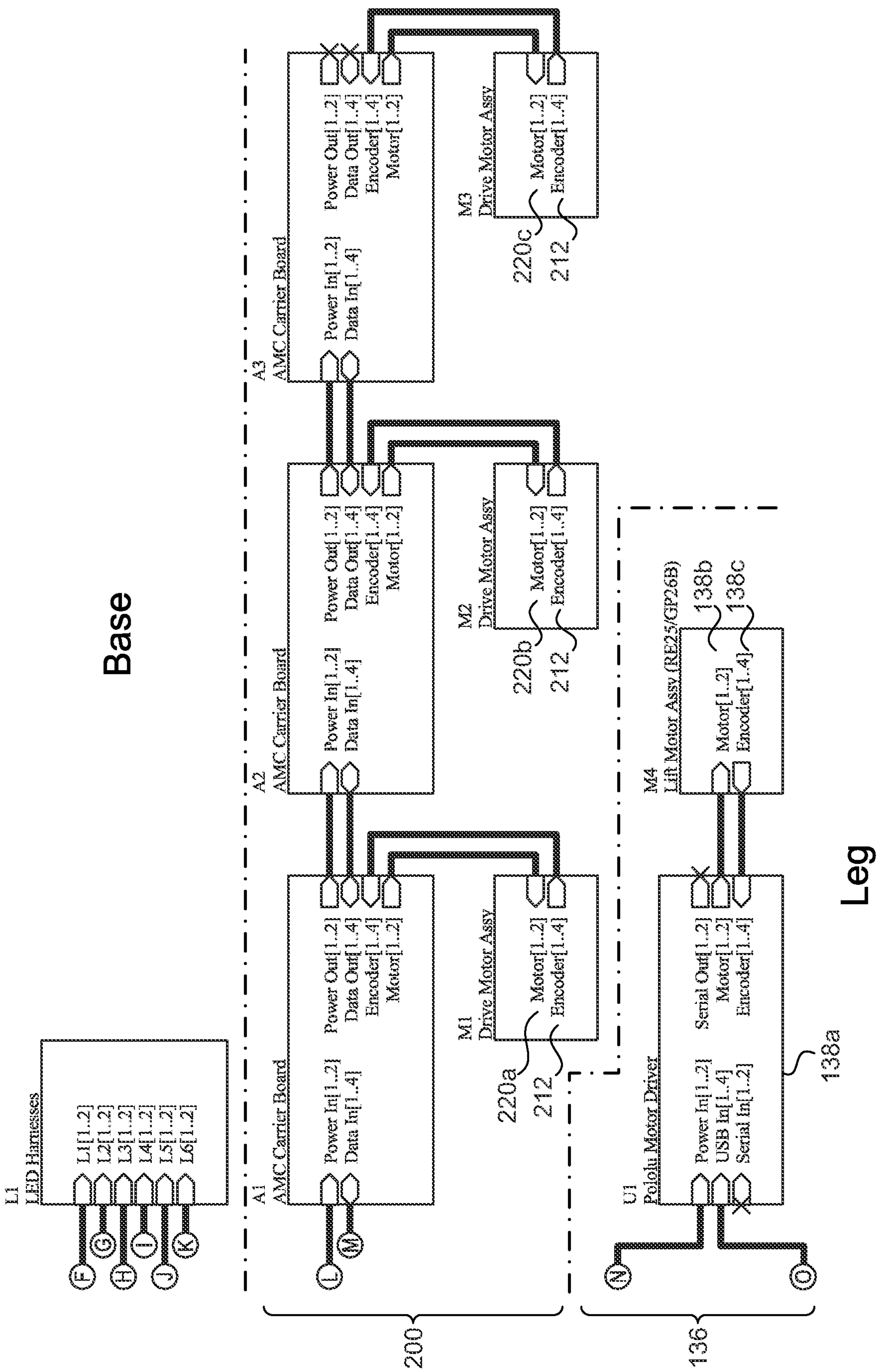
Base
Leg
L1
LED Harnesses
L1[1..2]
L2[1..2]
L3[1..2]
L4[1..2]
L5[1..2]
L6[1..2]
F
G
H
I
J
K
A1
AMC Carrier Board
Power In[1..2]
Data In[1..4]
Power Out[1..2]
Data Out[1..4]
Encoder[1..4]
Motor[1..2]
L
M
M1
Drive Motor Assy
Motor[1..2]
Encoder[1..4]
220a
212
A2
AMC Carrier Board
Power In[1..2]
Data In[1..4]
Power Out[1..2]
Data Out[1..4]
Encoder[1..4]
Motor[1..2]
M2
Drive Motor Assy
Motor[1..2]
Encoder[1..4]
220b
212
A3
AMC Carrier Board
Power In[1..2]
Data In[1..4]
Power Out[1..2]
Data Out[1..4]
Encoder[1..4]
Motor[1..2]
M3
Drive Motor Assy
Motor[1..2]
Encoder[1..4]
220c
212
U1
Pololu Motor Driver
Power In[1..2]
USB In[1..4]
Serial In[1..2]
Serial Out[1..2]
Motor[1..2]
Encoder[1..4]
N
O
138a
M4
Lift Motor Assy (RE25/GP26B)
Motor[1..2]
Encoder[1..4]
138b
138c
200
136

FIG. 8C

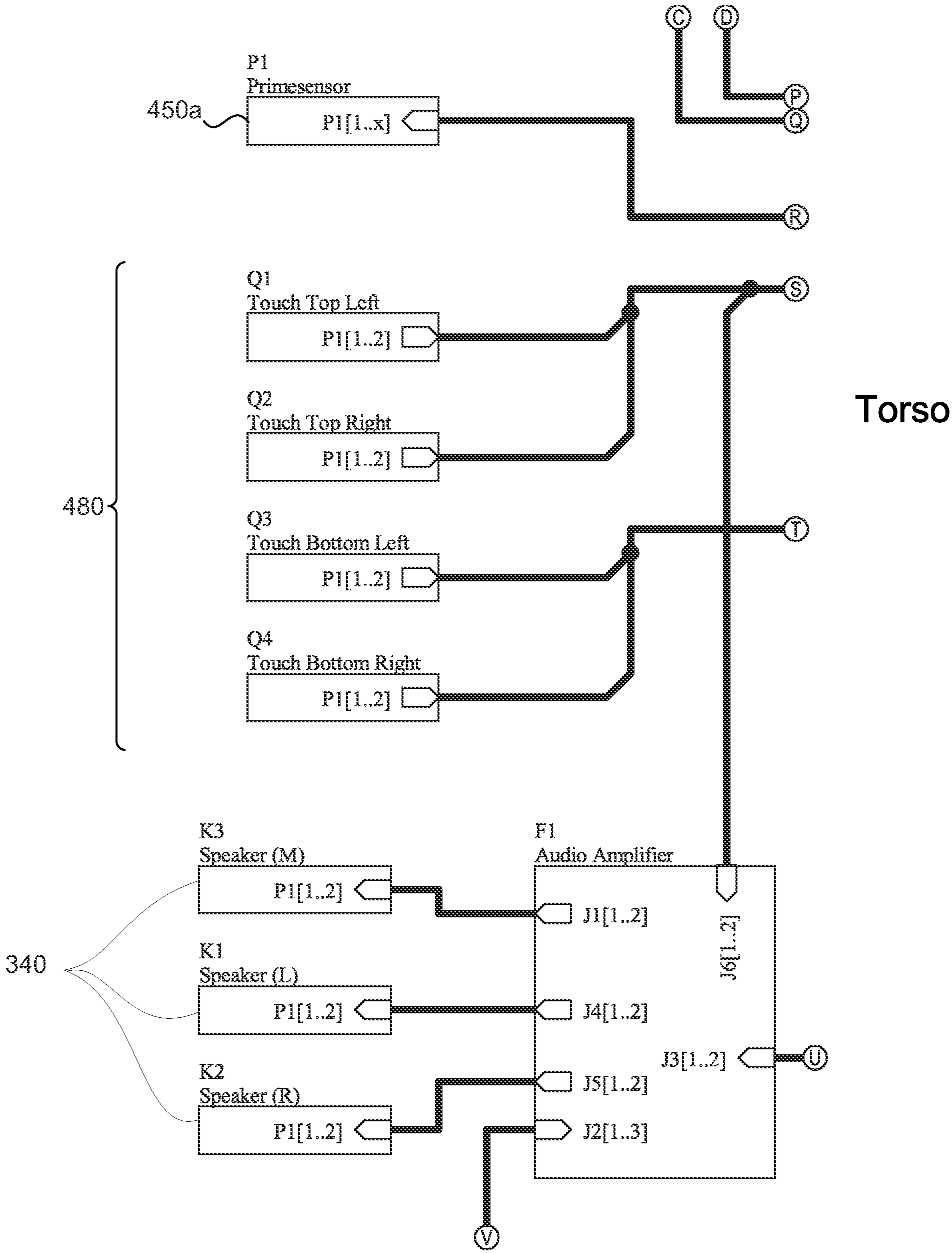
C
D
P1
Primesensor
450a
P1[1..x]
P
Q
R
Q1
Touch Top Left
P1[1..2]
S
Q2
Touch Top Right
P1[1..2]
Torso
480
Q3
Touch Bottom Left
P1[1..2]
T
Q4
Touch Bottom Right
P1[1..2]
K3
Speaker (M)
P1[1..2]
F1
Audio Amplifier
J1[1..2]
J6[1..2]
340
K1
Speaker (L)
P1[1..2]
J4[1..2]
J3[1..2]
U
K2
Speaker (R)
P1[1..2]
J5[1..2]
J2[1..3]
V

FIG. 8D

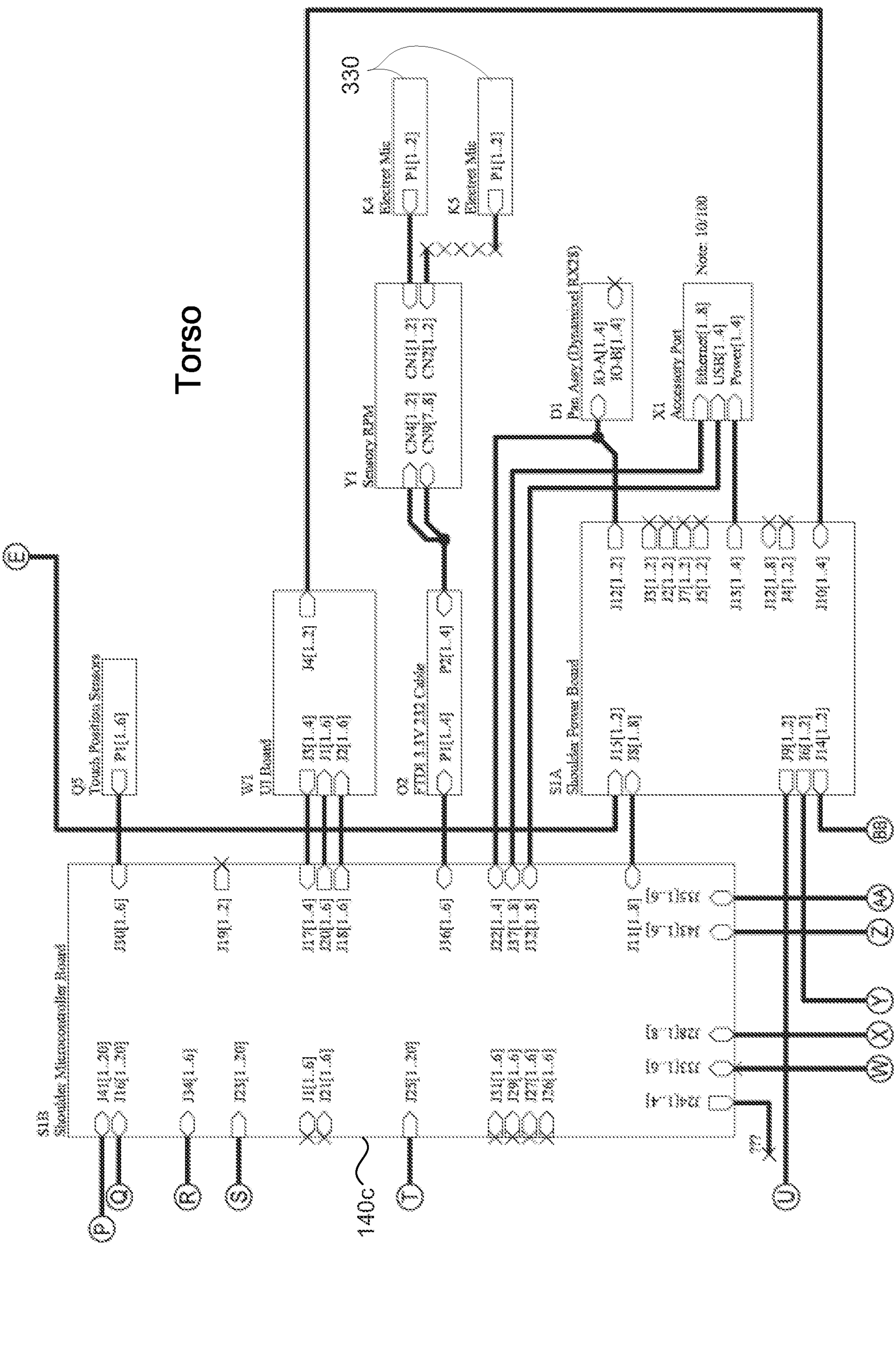

FIG. 8E

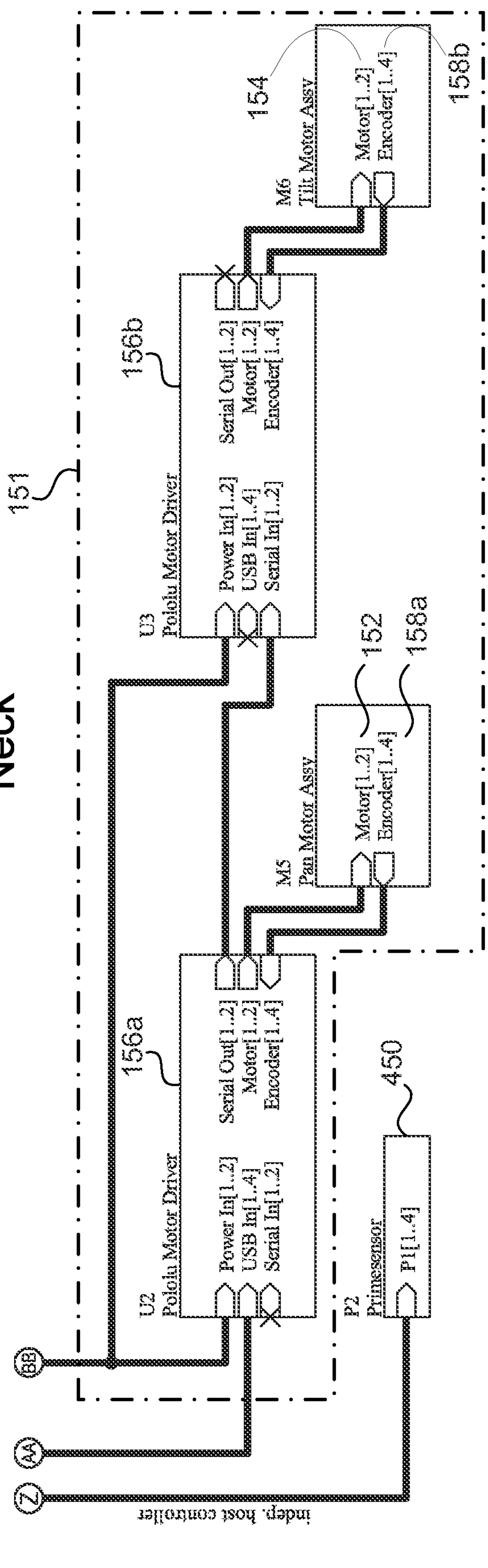
Neck
151
U3
Pololu Motor Driver
Power In[1..2]
USB In[1..4]
Serial In[1..2]
Serial Out[1..2]
Motor[1..2]
Encoder[1..4]
156b
M6
Tilt Motor Assy
Motor[1..2]
Encoder[1..4]
154
158b
M5
Pan Motor Assy
Motor[1..2]
Encoder[1..4]
152
158a
U2
Pololu Motor Driver
Power In[1..2]
USB In[1..4]
Serial In[1..2]
Serial Out[1..2]
Motor[1..2]
Encoder[1..4]
156a
P2
Primesensor
P1[1..4]
450
BB
AA
Z
indep. host controller

FIG. 8F

Head

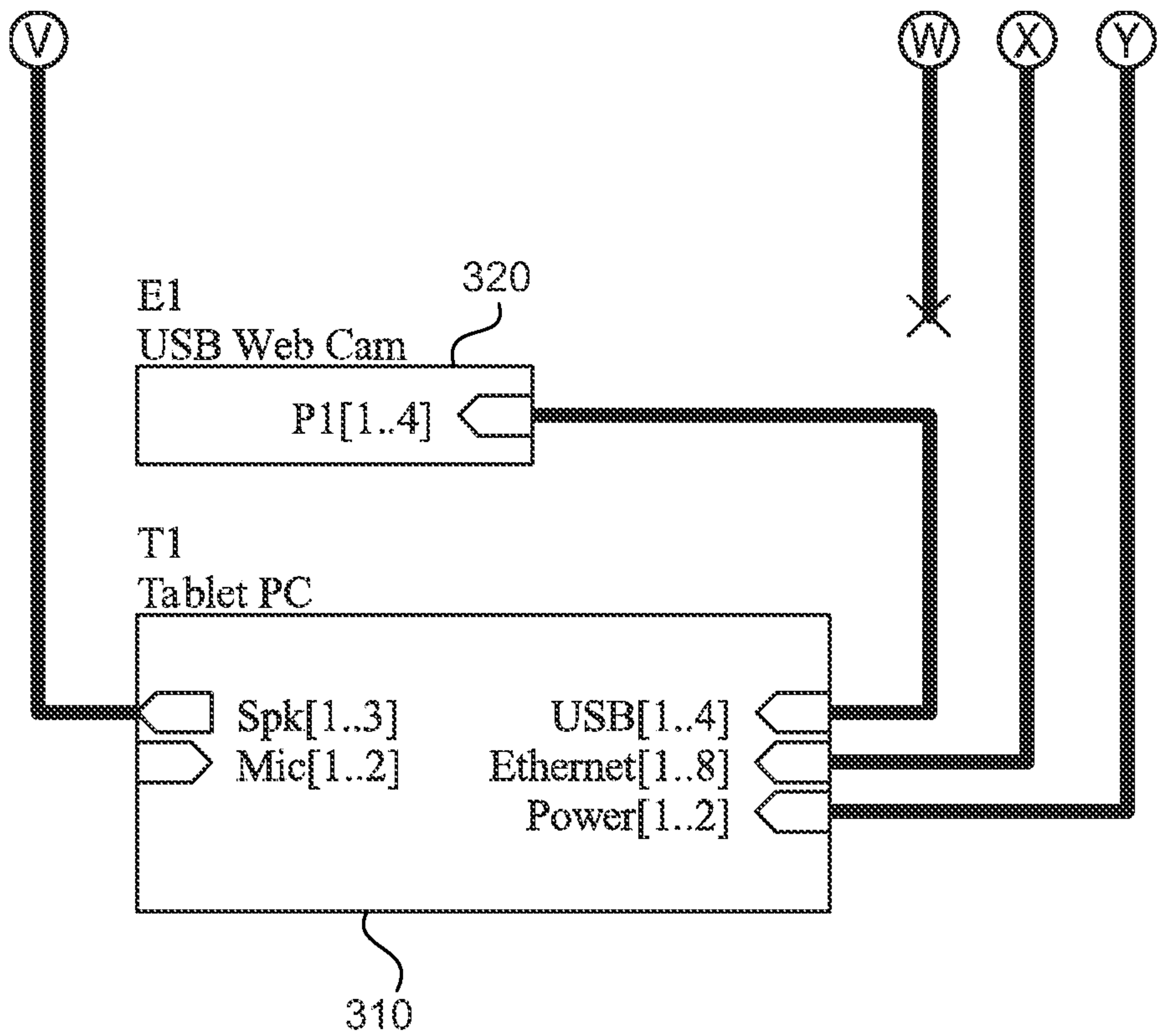
V
W
X
Y
E1
USB Web Cam
320
P1[1..4]
T1
Tablet PC
Spk[1..3]
Mic[1..2]
USB[1..4]
Ethernet[1..8]
Power[1..2]
310

FIG. 8G

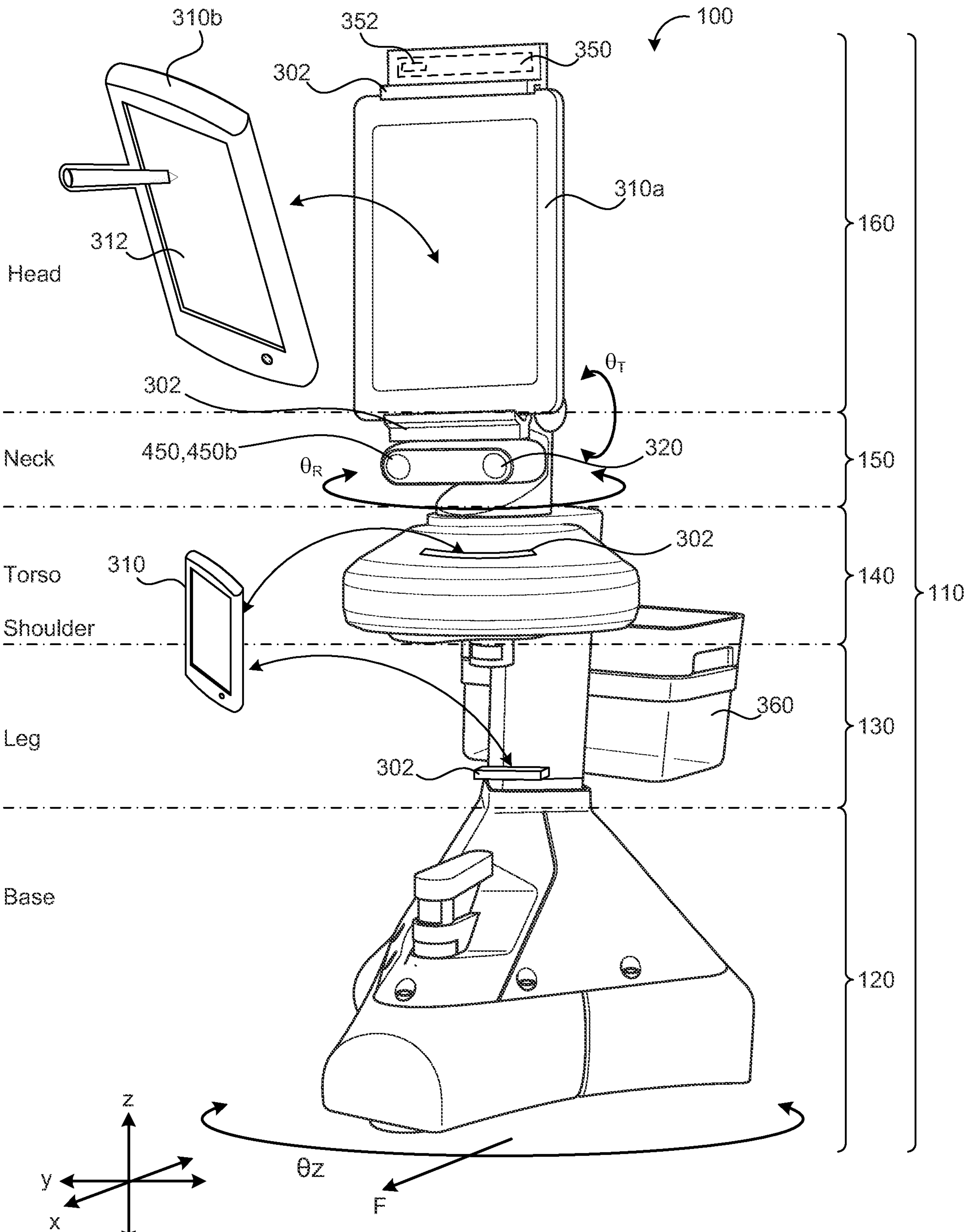
310b
352
100
350
302
310a
312
Head
160
θT
302
Neck
450,450b
320
θR
150
302
Torso
310
140
110
Shoulder
360
Leg
130
302
Base
120
z
y
x
θz
F

FIG. 9

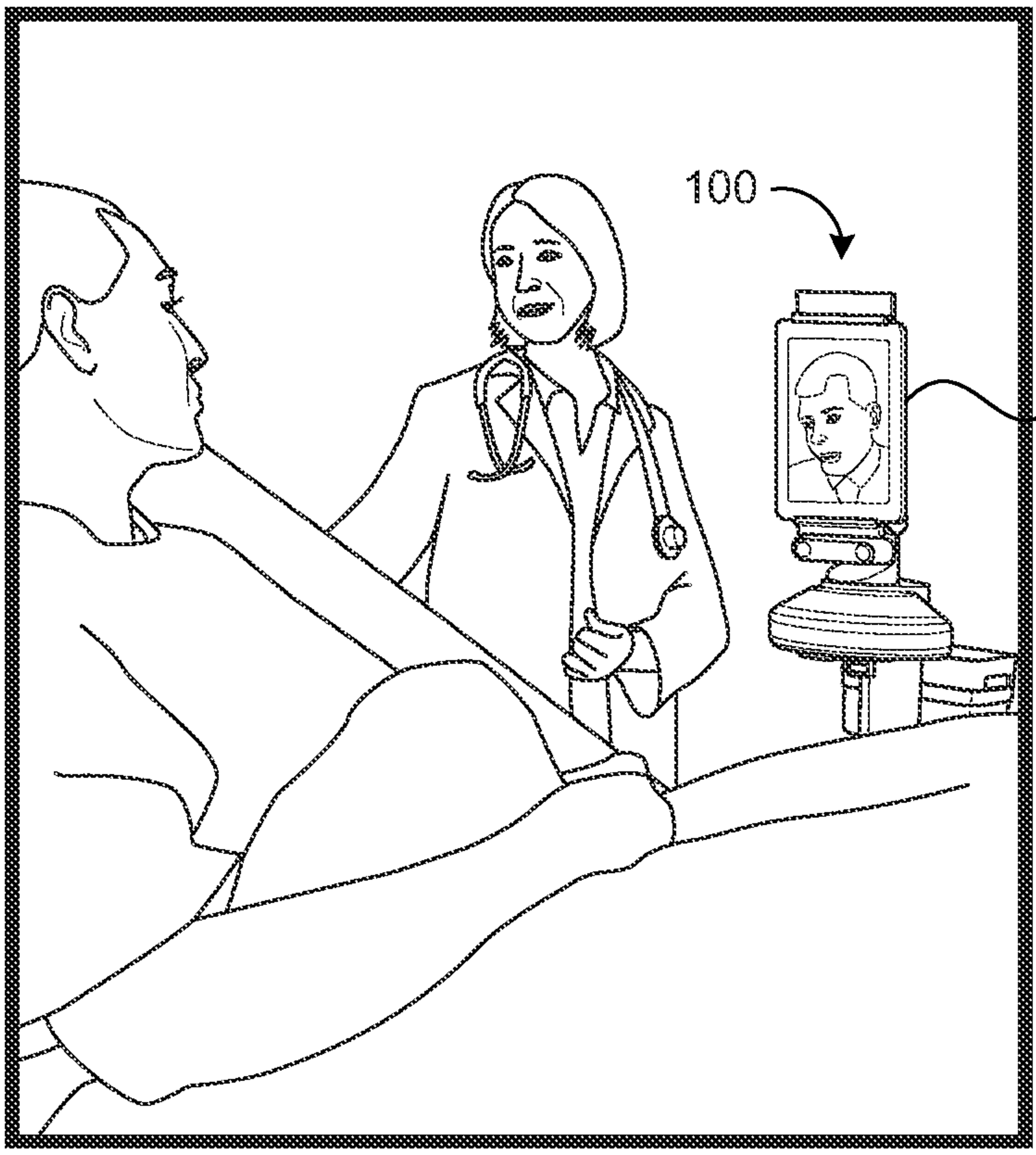

FIG. 10A

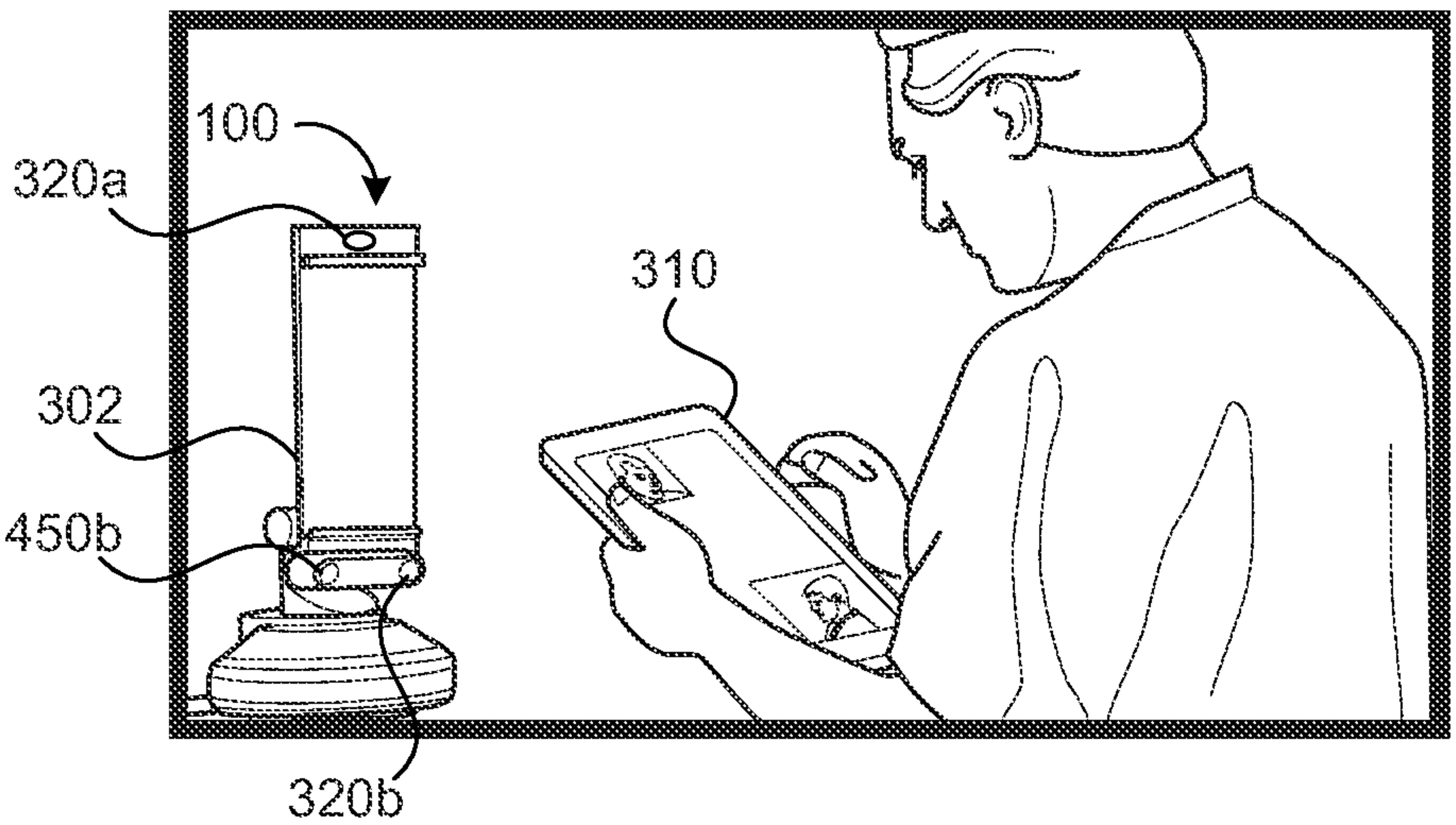

FIG. 10B

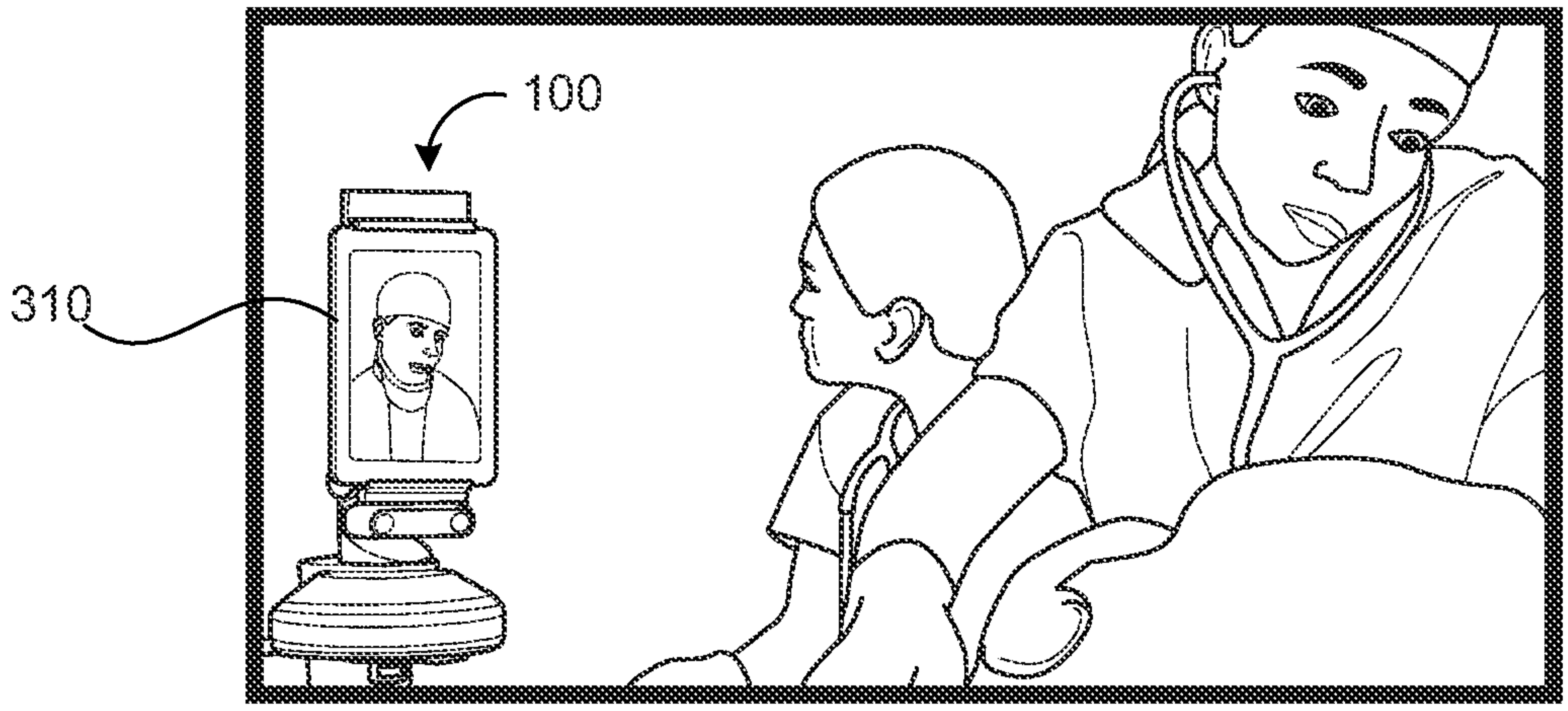

FIG. 10C

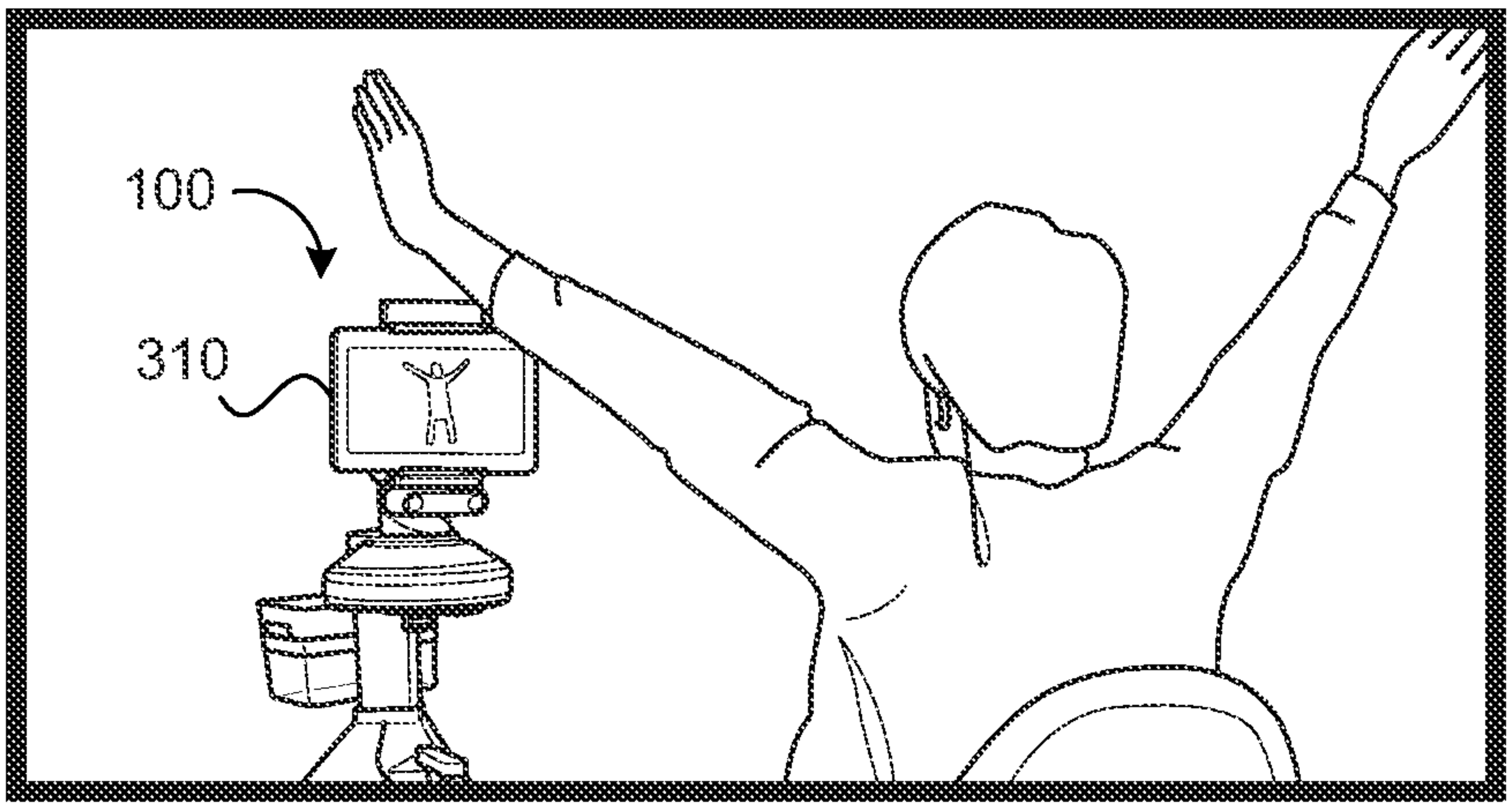

FIG. 10D

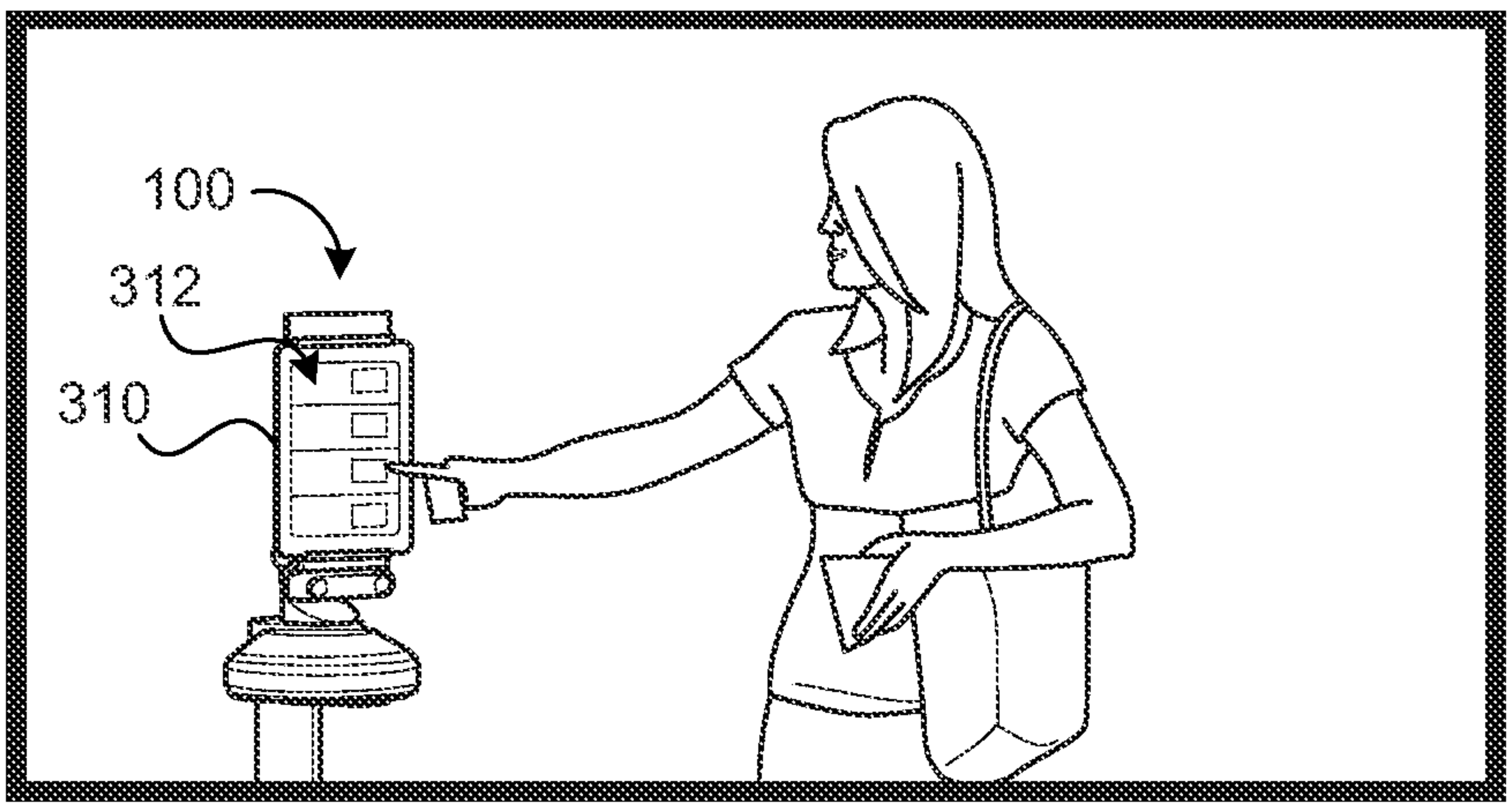

FIG. 10E

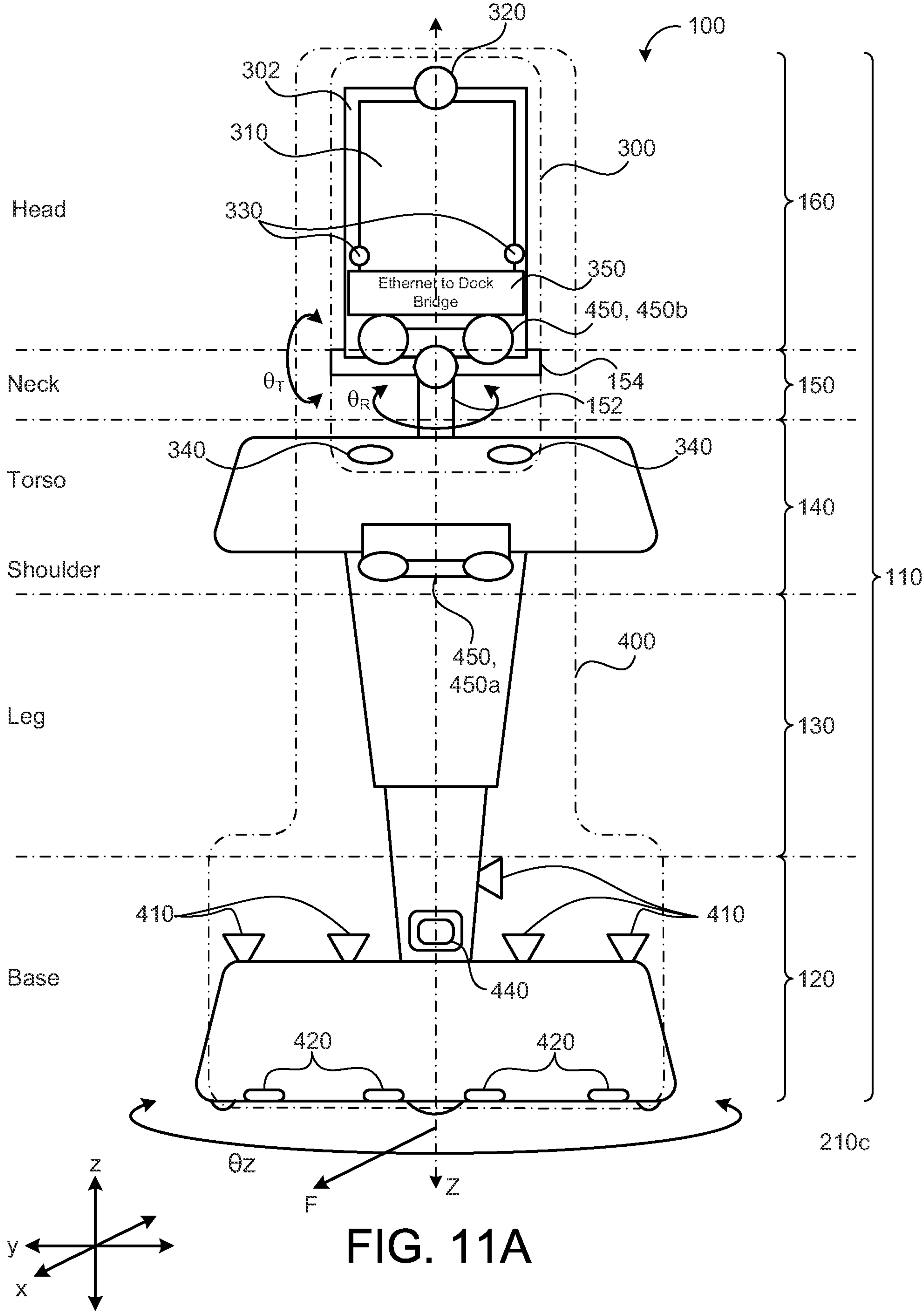
320
100
302
310
300
Head
330
160
Ethernet to Dock Bridge
350
450, 450b
Neck
θ_T
θ_R
154
152
150
Torso
340
340
140
Shoulder
110
450, 450a
400
Leg
130
410
410
Base
440
120
420
420
210c
θz
z
F
Z
y
x

FIG. 11A

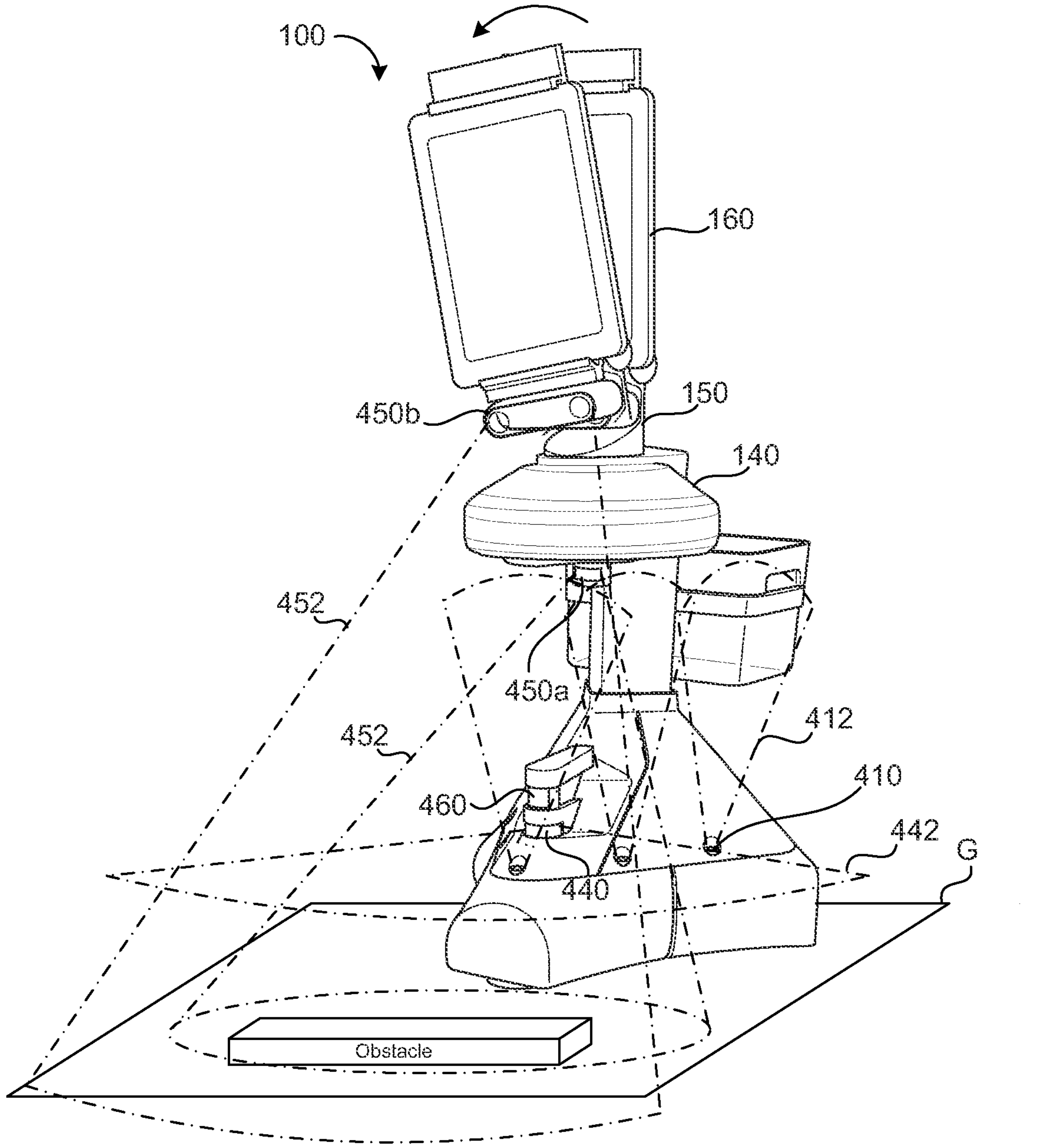
100
160
150
450b
140
452
450a
452
412
410
460
442
G
440
Obstacle

FIG. 11B

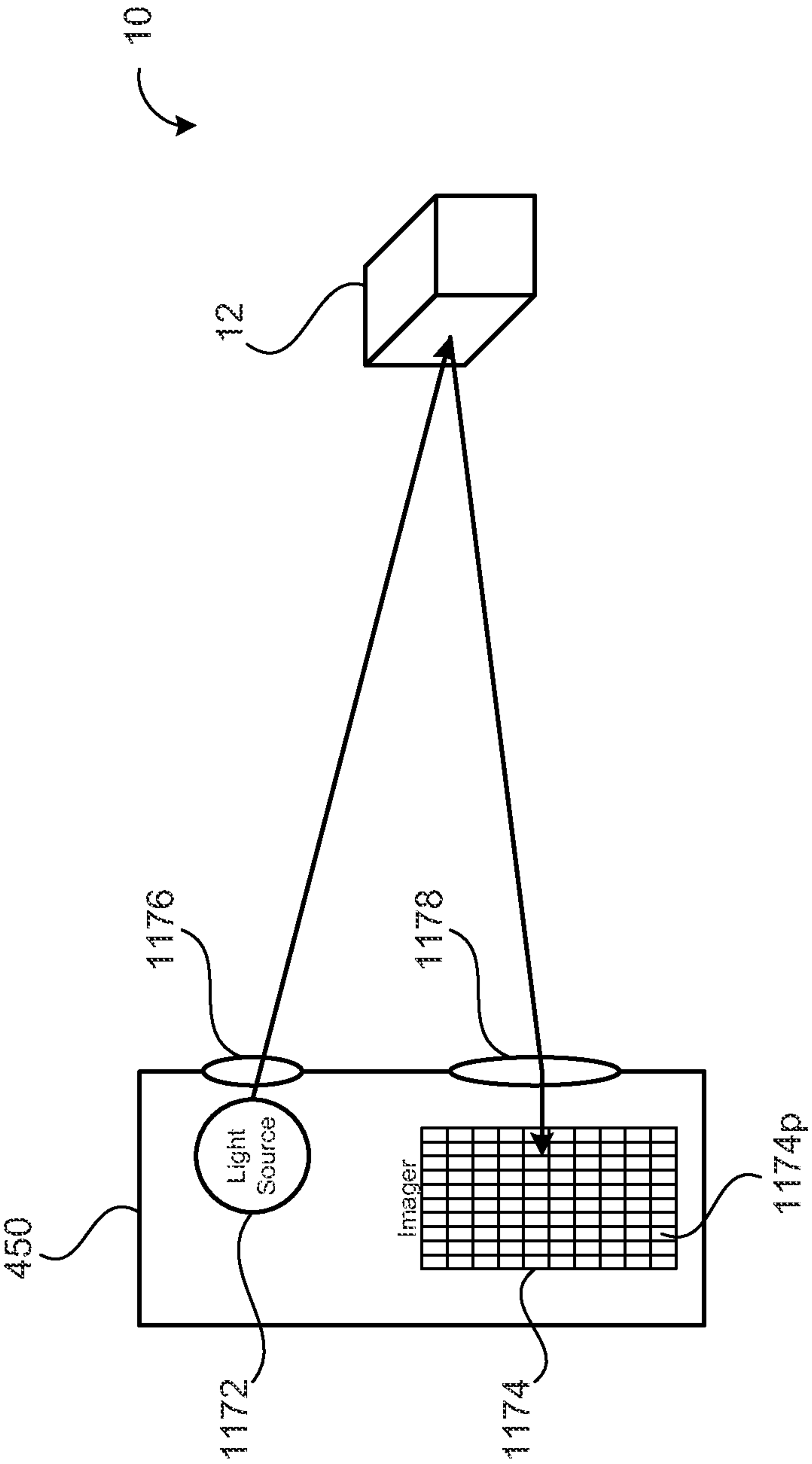

FIG. 12A

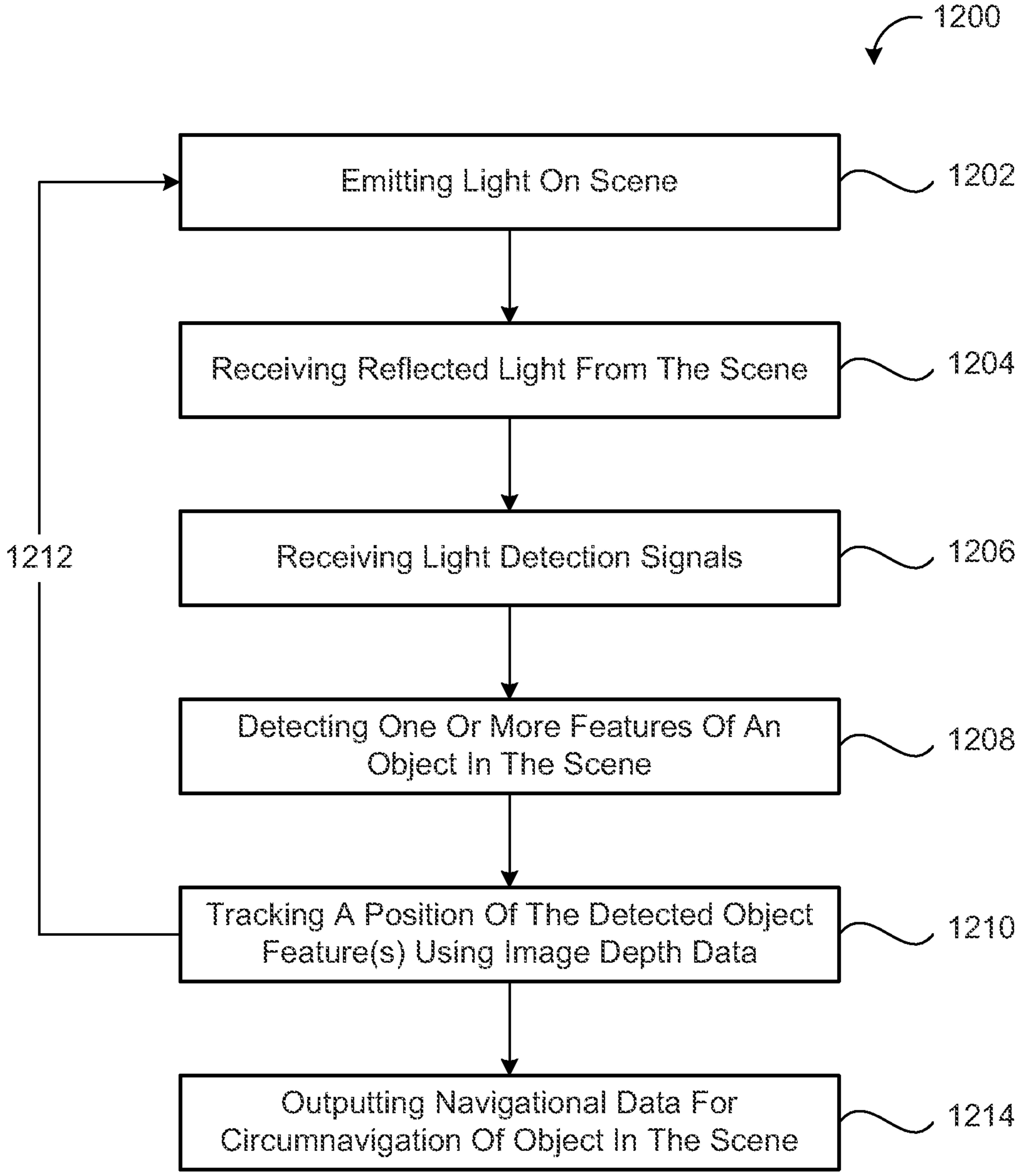
1200
Emitting Light On Scene
1202
Receiving Reflected Light From The Scene
1204
Receiving Light Detection Signals
1206
Detecting One Or More Features Of An Object In The Scene
1208
Tracking A Position Of The Detected Object Feature(s) Using Image Depth Data
1210
1212
Outputting Navigational Data For Circumnavigation Of Object In The Scene
1214

FIG. 12B

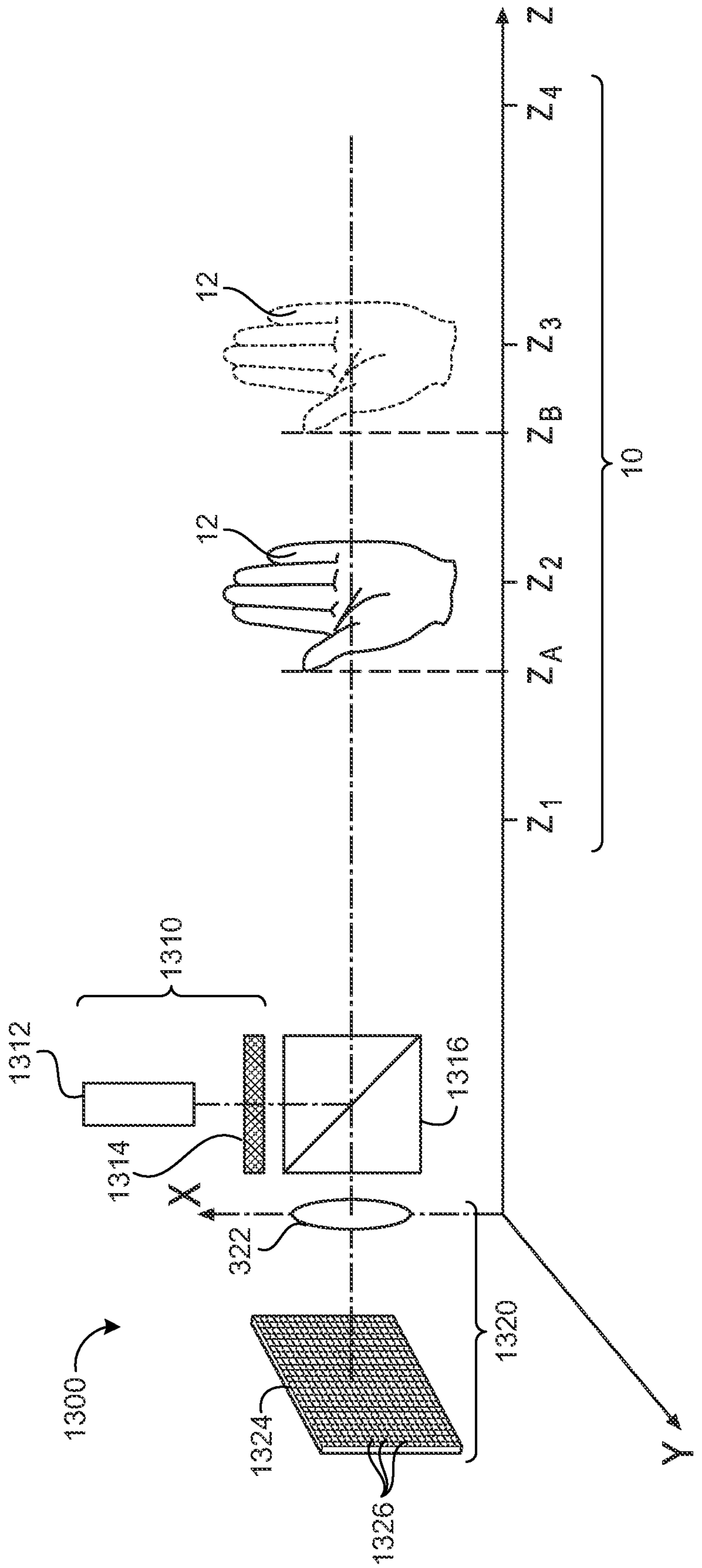
Z
Z_4
12
Z_3
Z_B
10
12
Z_2
Z_A
Z_1
1310
1312
1316
1314
X
322
1300
1320
1324
1326
Y

FIG. 12C

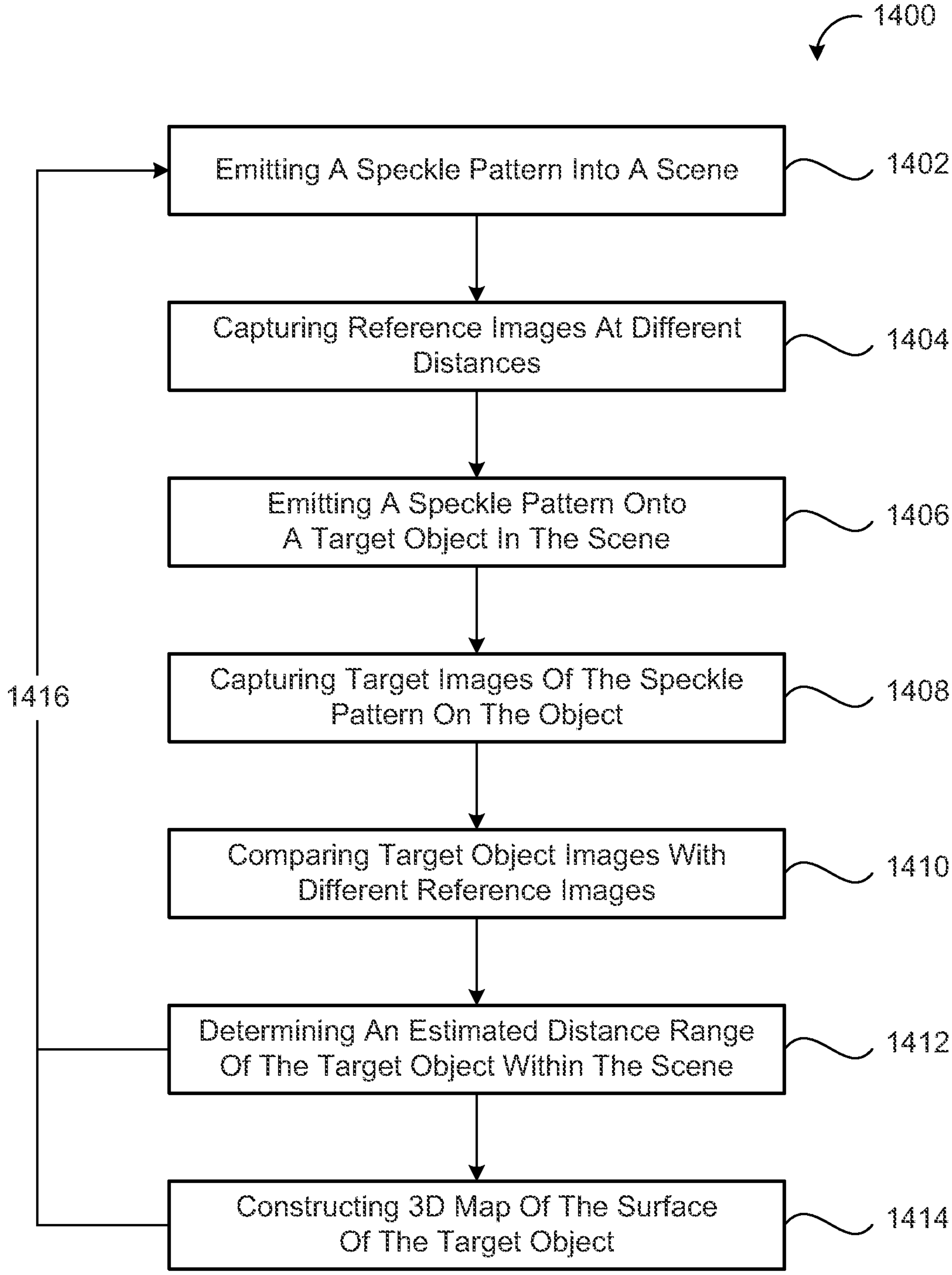
1400
Emitting A Speckle Pattern Into A Scene
1402
Capturing Reference Images At Different Distances
1404
Emitting A Speckle Pattern Onto A Target Object In The Scene
1406
Capturing Target Images Of The Speckle Pattern On The Object
1408
Comparing Target Object Images With Different Reference Images
1410
Determining An Estimated Distance Range Of The Target Object Within The Scene
1412
Constructing 3D Map Of The Surface Of The Target Object
1414
1416

FIG. 12D

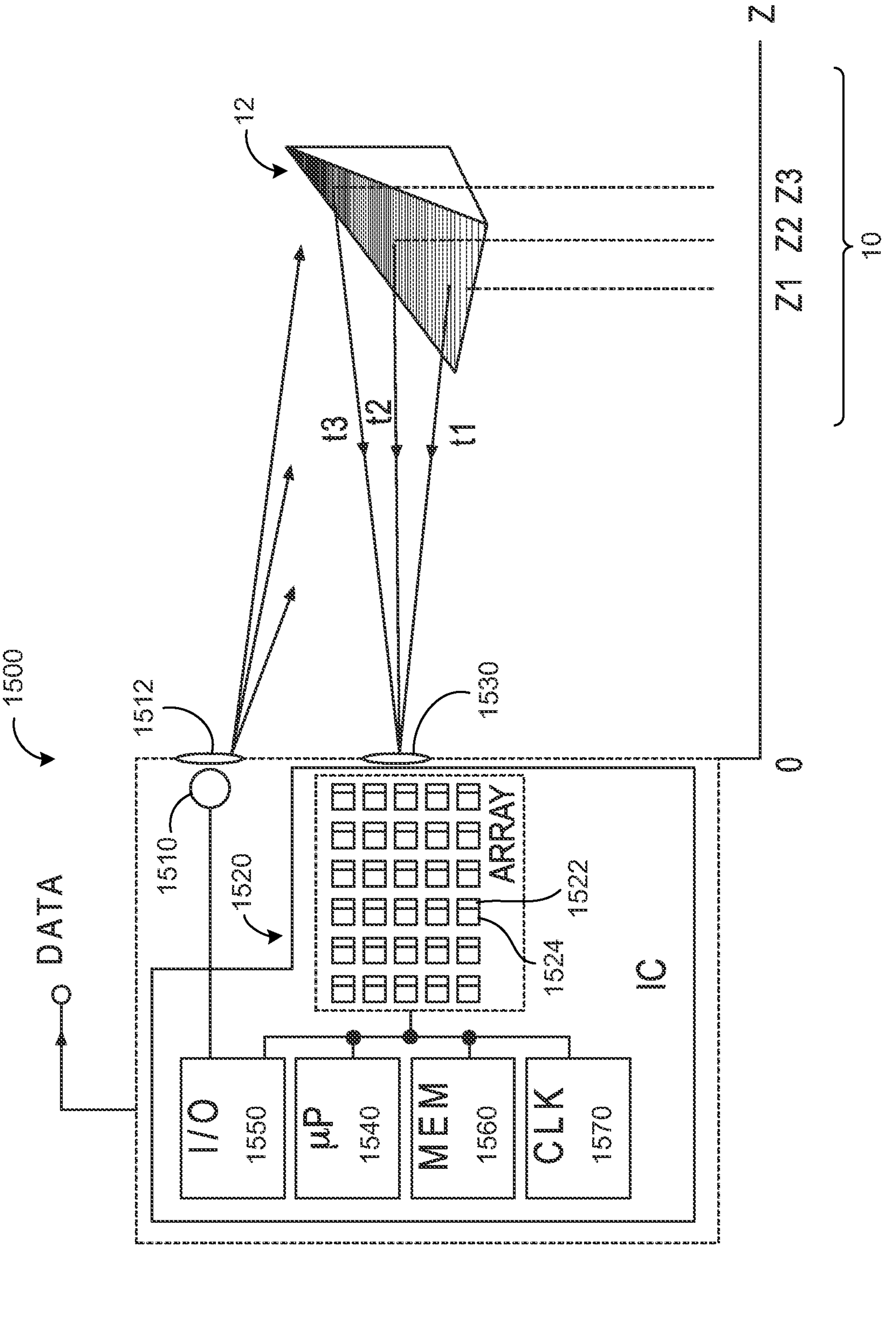
1500
DATA
1512
1510
1520
1530
ARRAY
1522
1524
IC
I/O
1550
µP
1540
MEM
1560
CLK
1570
0
Z
12
t1
t2
t3
Z1 Z2 Z3
10

FIG. 12E

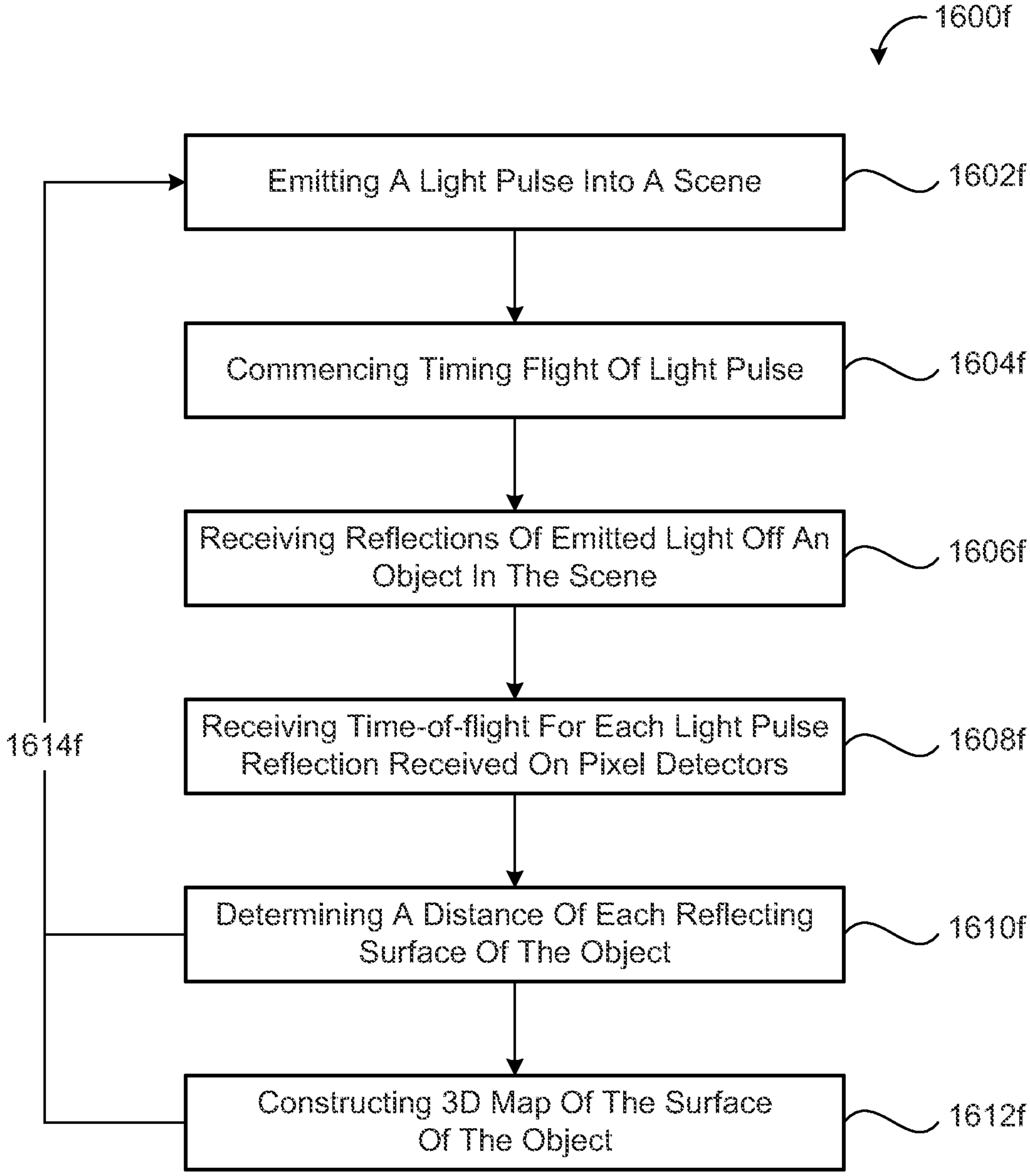
1600f
Emitting A Light Pulse Into A Scene
1602f
Commencing Timing Flight Of Light Pulse
1604f
Receiving Reflections Of Emitted Light Off An Object In The Scene
1606f
Receiving Time-of-flight For Each Light Pulse Reflection Received On Pixel Detectors
1608f
Determining A Distance Of Each Reflecting Surface Of The Object
1610f
Constructing 3D Map Of The Surface Of The Object
1612f
1614f

FIG. 12F

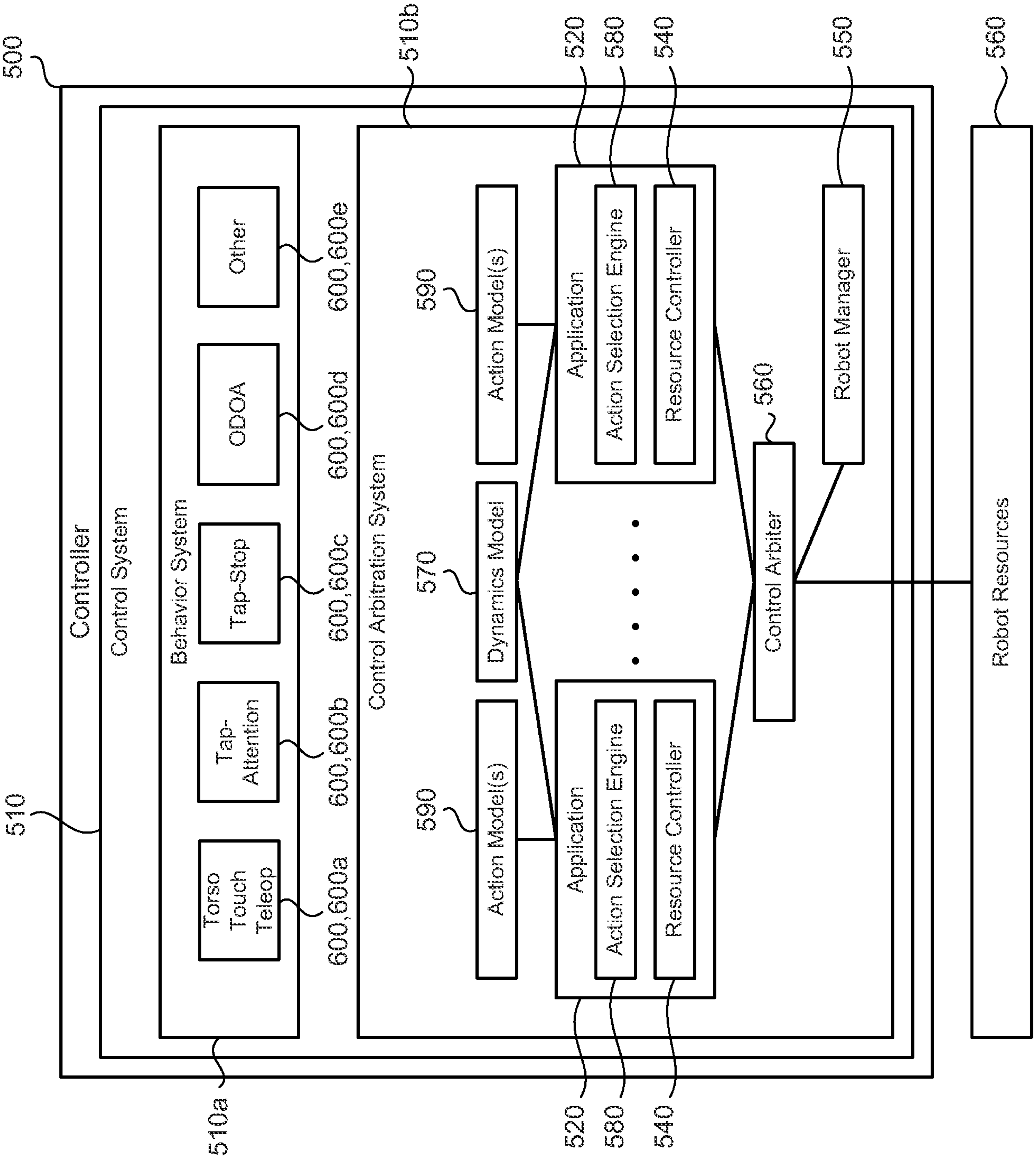
500
510
Controller
Control System
Behavior System
510a
Torso Touch Teleop
600,600a
Tap-Attention
600,600b
Tap-Stop
600,600c
ODOA
600,600d
Other
600,600e
Control Arbitration System
510b
590
Action Model(s)
570
Dynamics Model
520
580
540
Application
Action Selection Engine
Resource Controller
560
Control Arbiter
550
Robot Manager
560
Robot Resources

FIG. 13

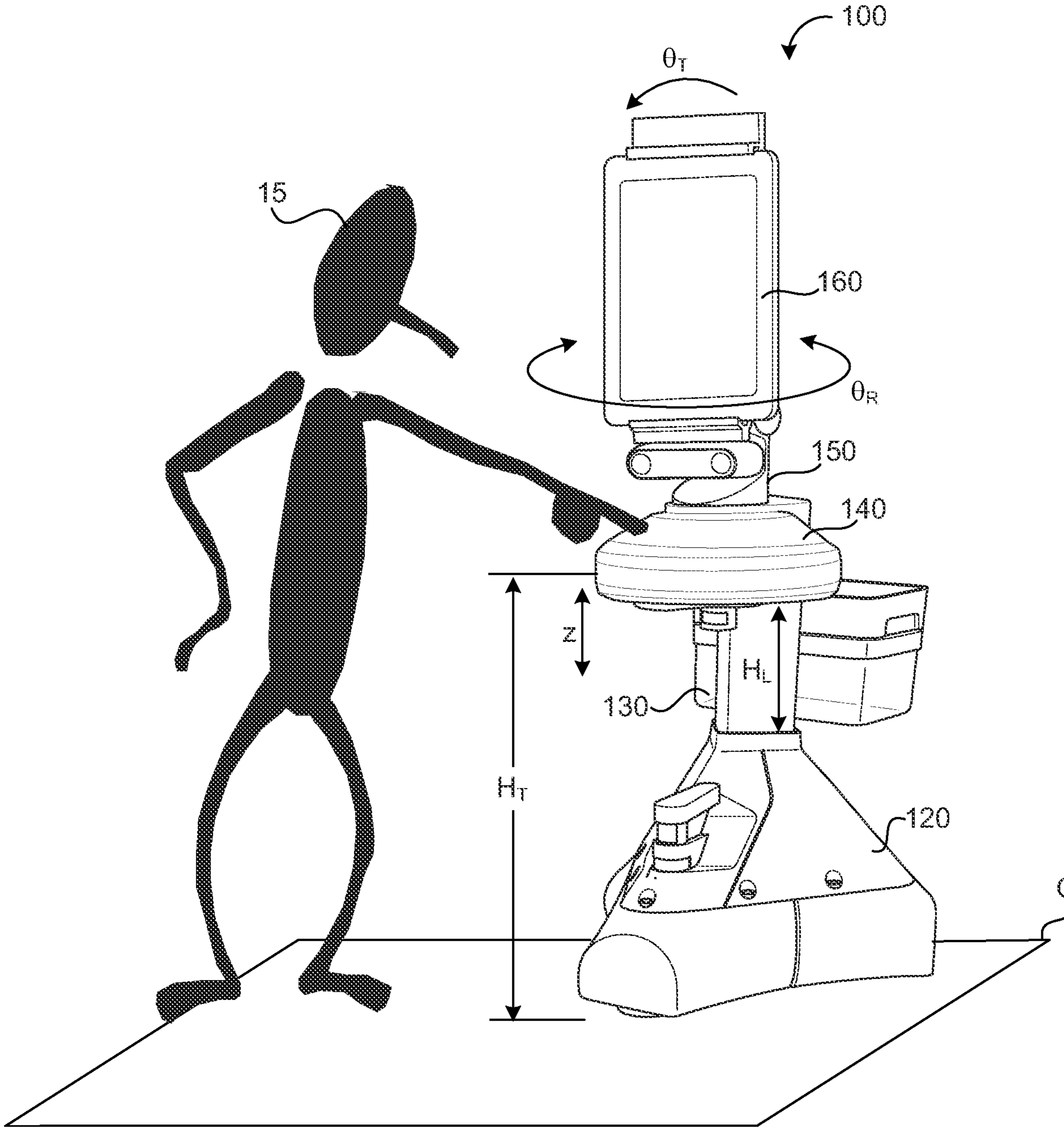

FIG. 14

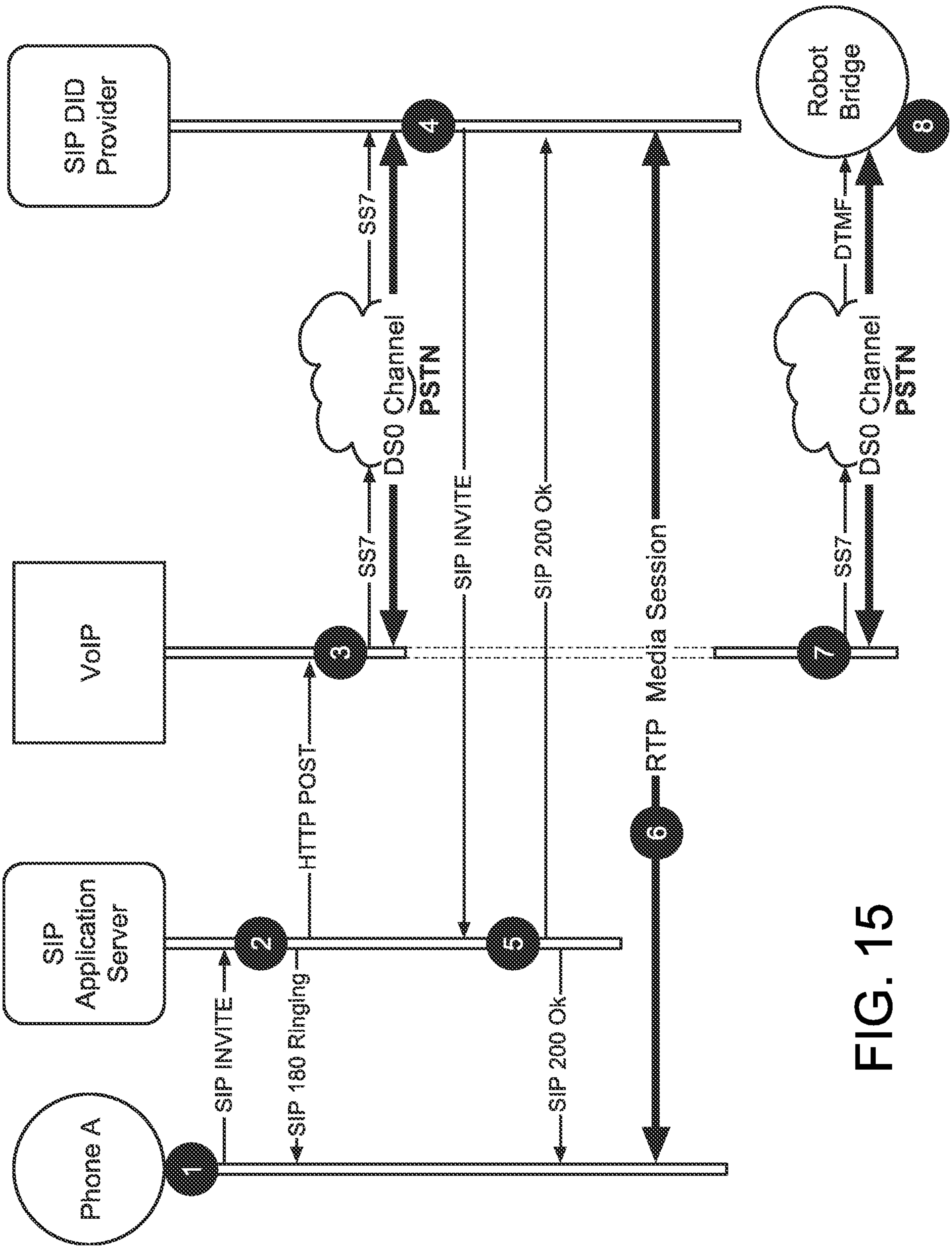
Phone A
SIP Application Server
VoIP
SIP DID Provider
Robot Bridge
SIP INVITE
SIP 180 Ringing
SIP 200 Ok
HTTP POST
SS7
DS0 Channel
PSTN
SIP INVITE
SIP 200 Ok
RTP Media Session
DTMF
1
2
3
4
5
6
7
8

FIG. 15

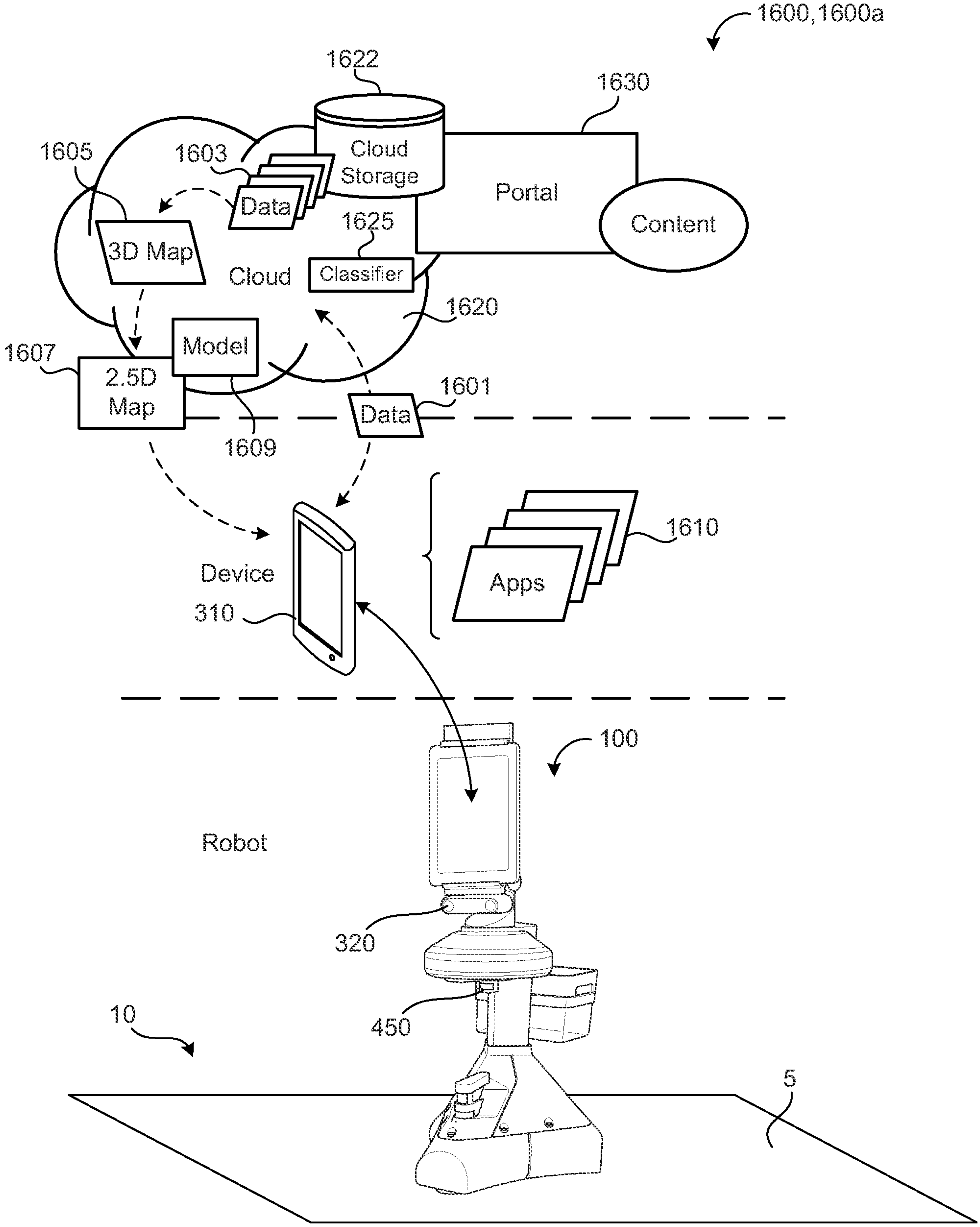
1600,1600a
1622
Cloud Storage
1630
Portal
Content
1605
1603
Data
3D Map
Cloud
1625
Classifier
1620
1607
2.5D Map
Model
1609
Data
1601
Device
310
Apps
1610
100
Robot
320
450
10
5

FIG. 16A

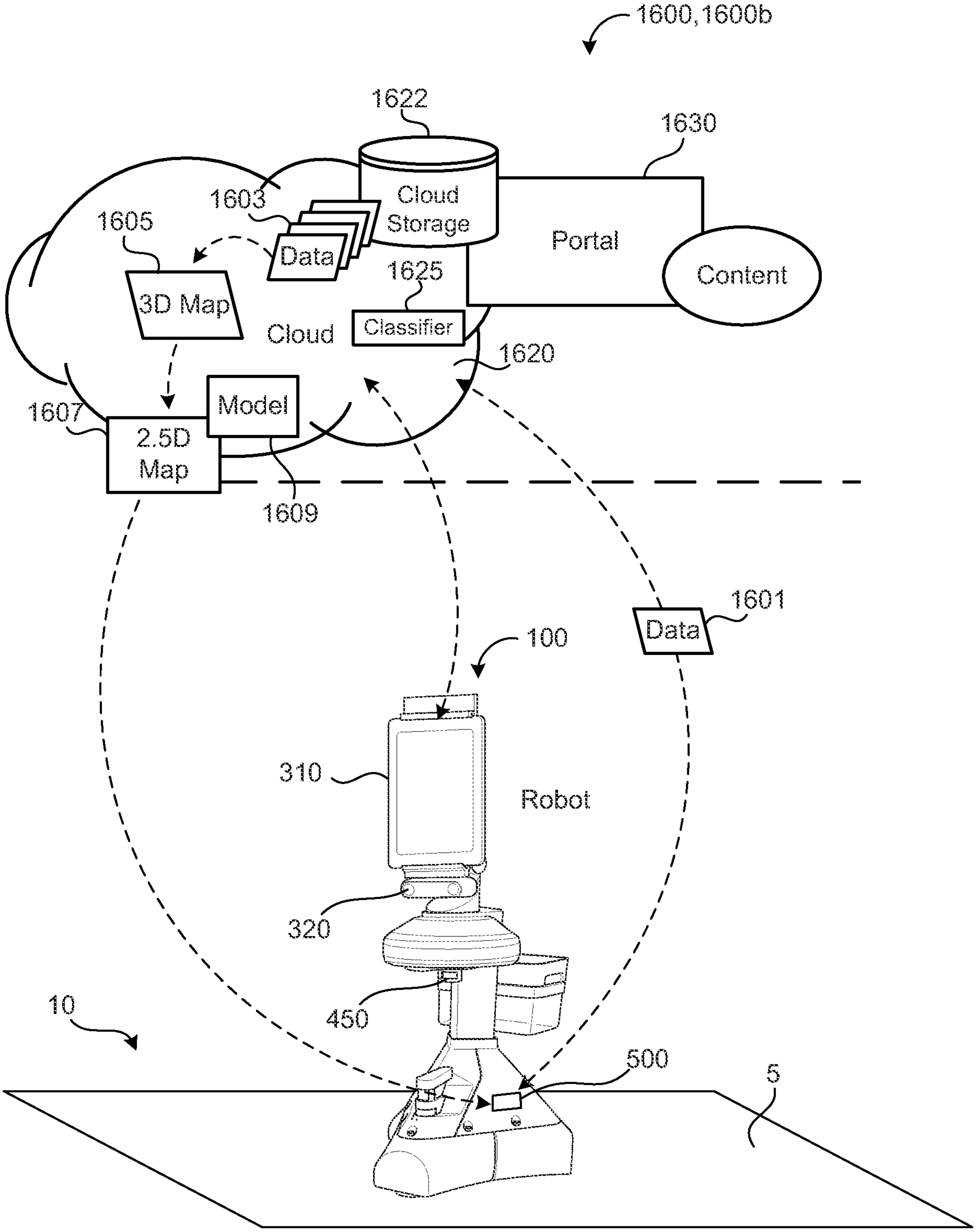
1600,1600b
1622
1630
Cloud
Storage
1603
1605
Data
Portal
Content
1625
3D Map
Classifier
Cloud
1620
Model
1607
2.5D
Map
1609
Data
1601
100
310
Robot
320
450
10
500
5

FIG. 16B

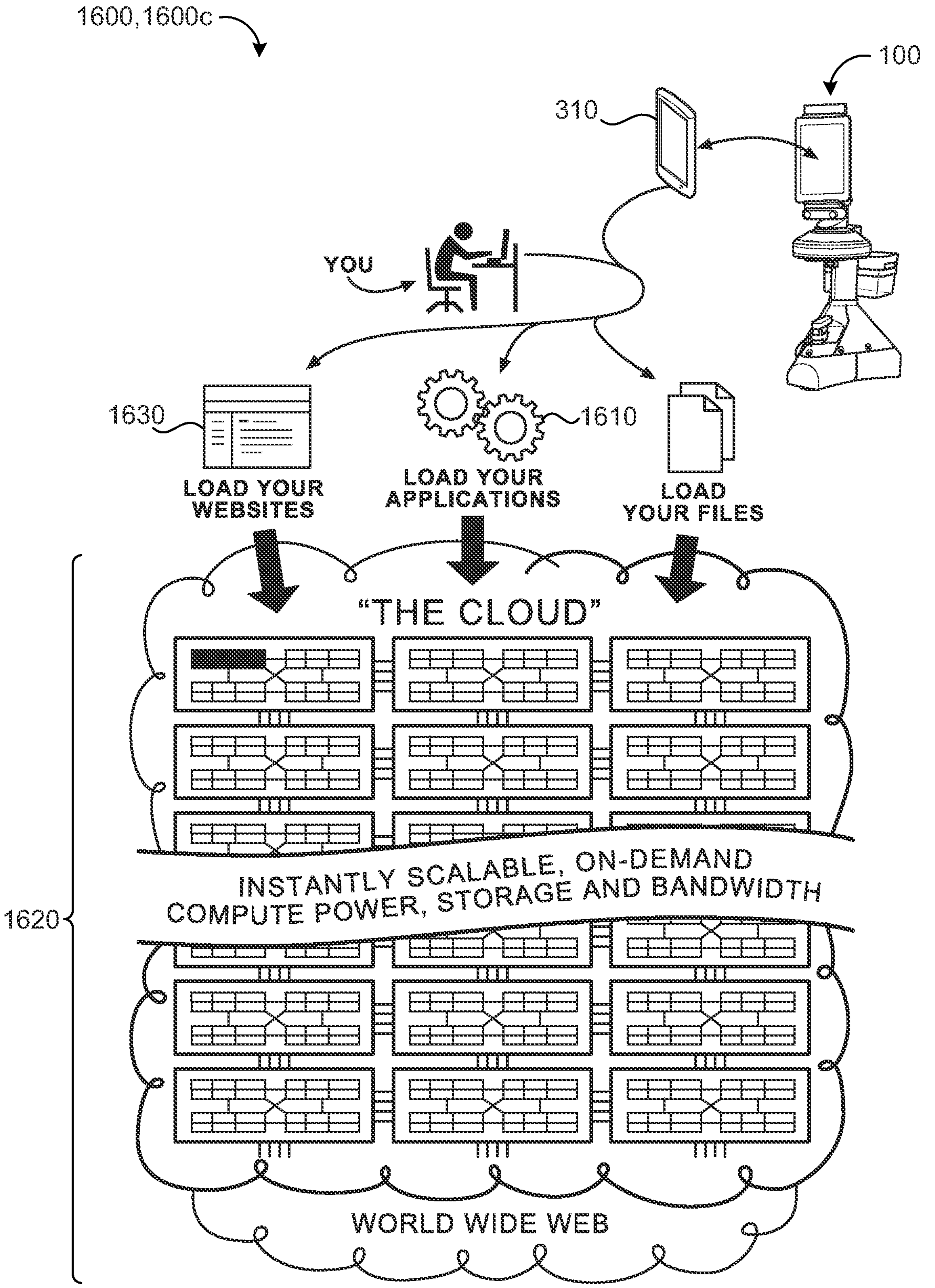
1600,1600c
100
310
YOU
1630
1610
LOAD YOUR WEBSITES
LOAD YOUR APPLICATIONS
LOAD YOUR FILES
"THE CLOUD"
INSTANTLY SCALABLE, ON-DEMAND COMPUTE POWER, STORAGE AND BANDWIDTH
1620
WORLD WIDE WEB

FIG. 16C

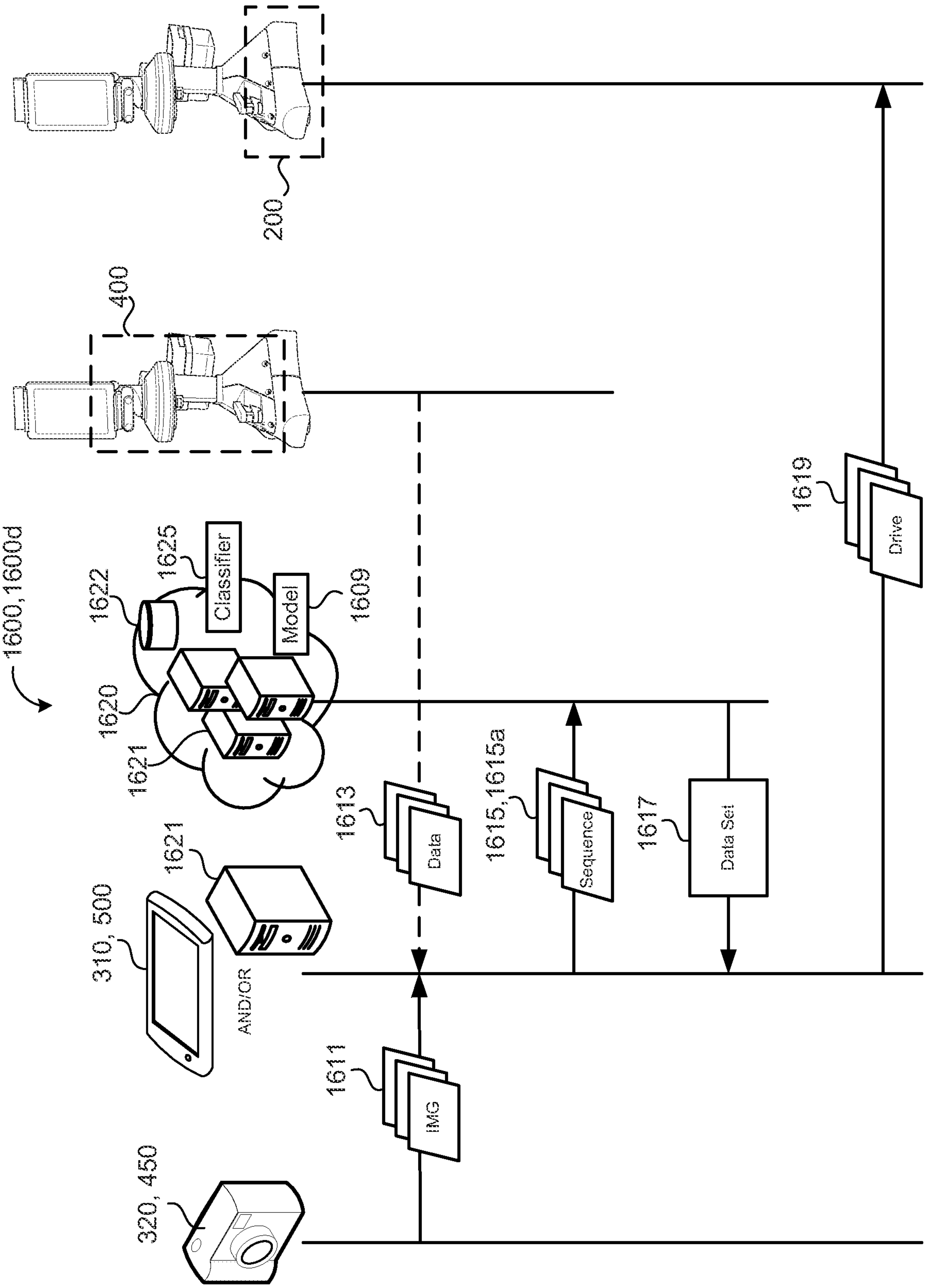
1600,1600d
200
400
1625
Classifier
1622
Model
1609
1620
1621
1613
Data
1615,1615a
Sequence
1617
Data Set
1619
Drive
310, 500
1621
AND/OR
1611
IMG
320, 450

FIG. 16D

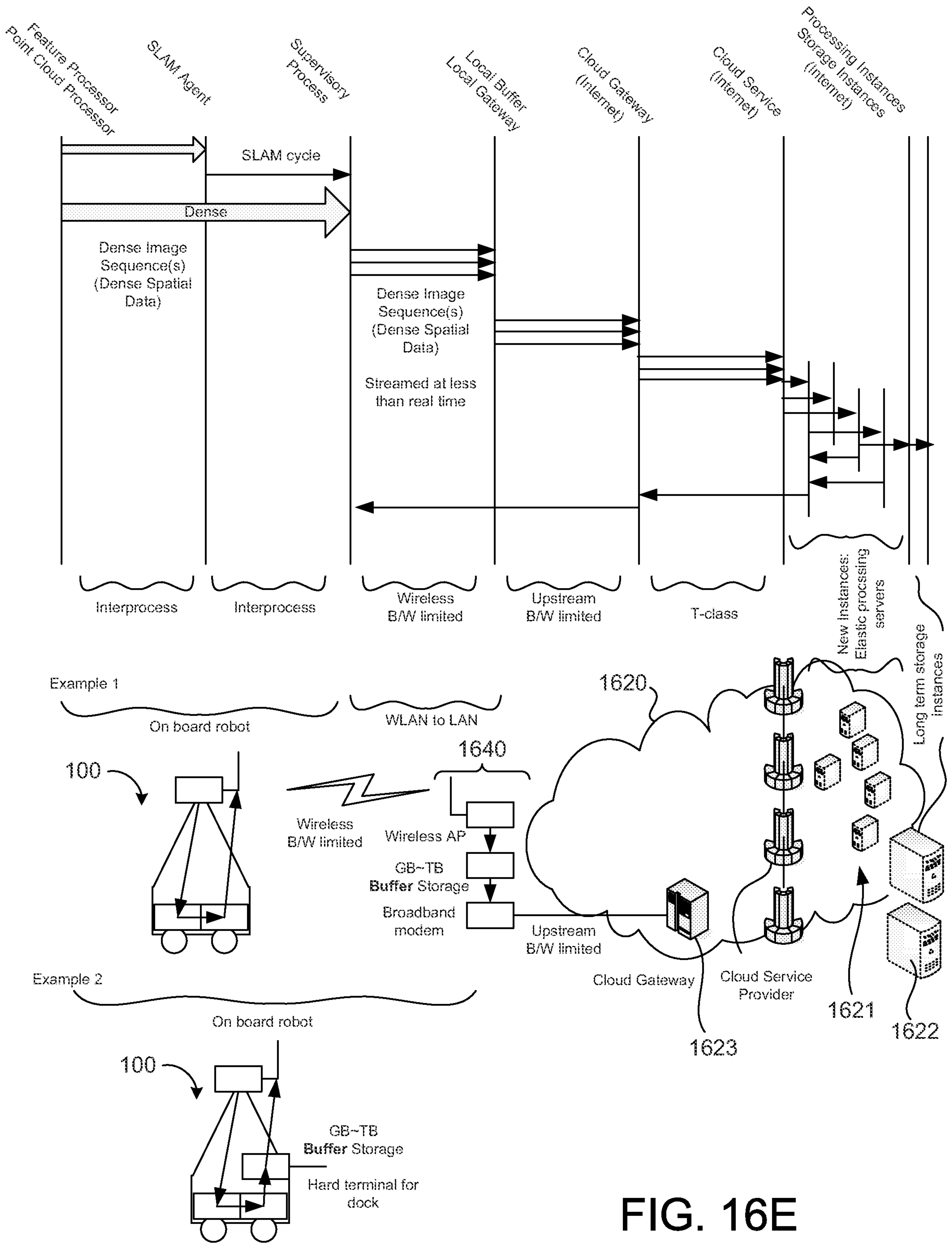
Feature Processor
Point Cloud Processor
SLAM Agent
Supervisory Process
Local Buffer
Local Gateway
Cloud Gateway (Internet)
Cloud Service (Internet)
Processing Instances
Storage Instances
(Internet)
SLAM cycle
Dense
Dense Image Sequence(s) (Dense Spatial Data)
Dense Image Sequence(s) (Dense Spatial Data)
Streamed at less than real time
Interprocess
Interprocess
Wireless B/W limited
Upstream B/W limited
T-class
New Instances: Elastic processing servers
Long term storage instances
Example 1
On board robot
WLAN to LAN
1620
1640
100
Wireless B/W limited
Wireless AP
GB~TB Buffer Storage
Broadband modem
Upstream B/W limited
Cloud Gateway
Cloud Service Provider
1621
1622
1623
Example 2
On board robot
100
GB~TB Buffer Storage
Hard terminal for dock

FIG. 16E

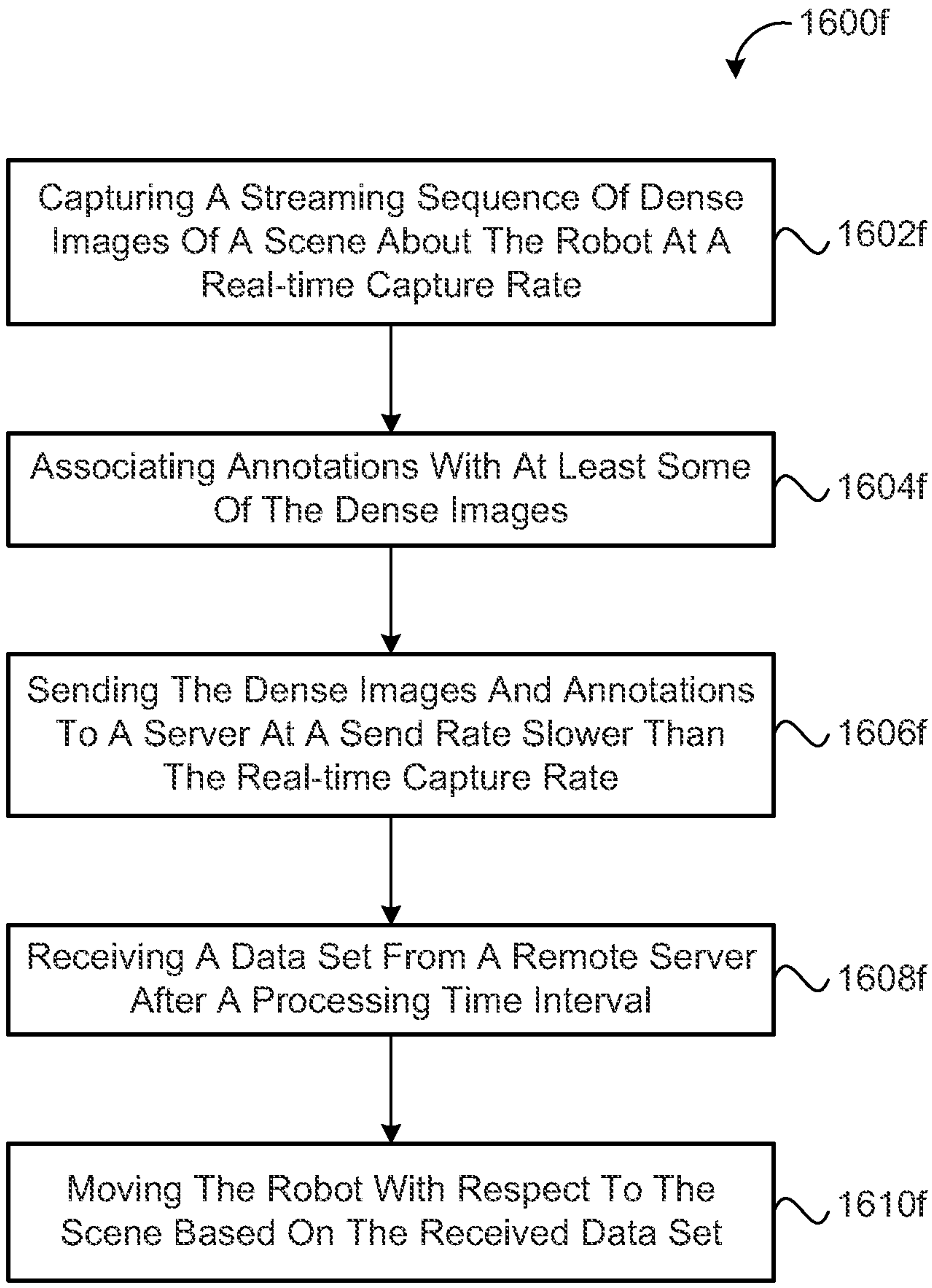
1600f
Capturing A Streaming Sequence Of Dense Images Of A Scene About The Robot At A Real-time Capture Rate
1602f
Associating Annotations With At Least Some Of The Dense Images
1604f
Sending The Dense Images And Annotations To A Server At A Send Rate Slower Than The Real-time Capture Rate
1606f
Receiving A Data Set From A Remote Server After A Processing Time Interval
1608f
Moving The Robot With Respect To The Scene Based On The Received Data Set
1610f

FIG. 16F

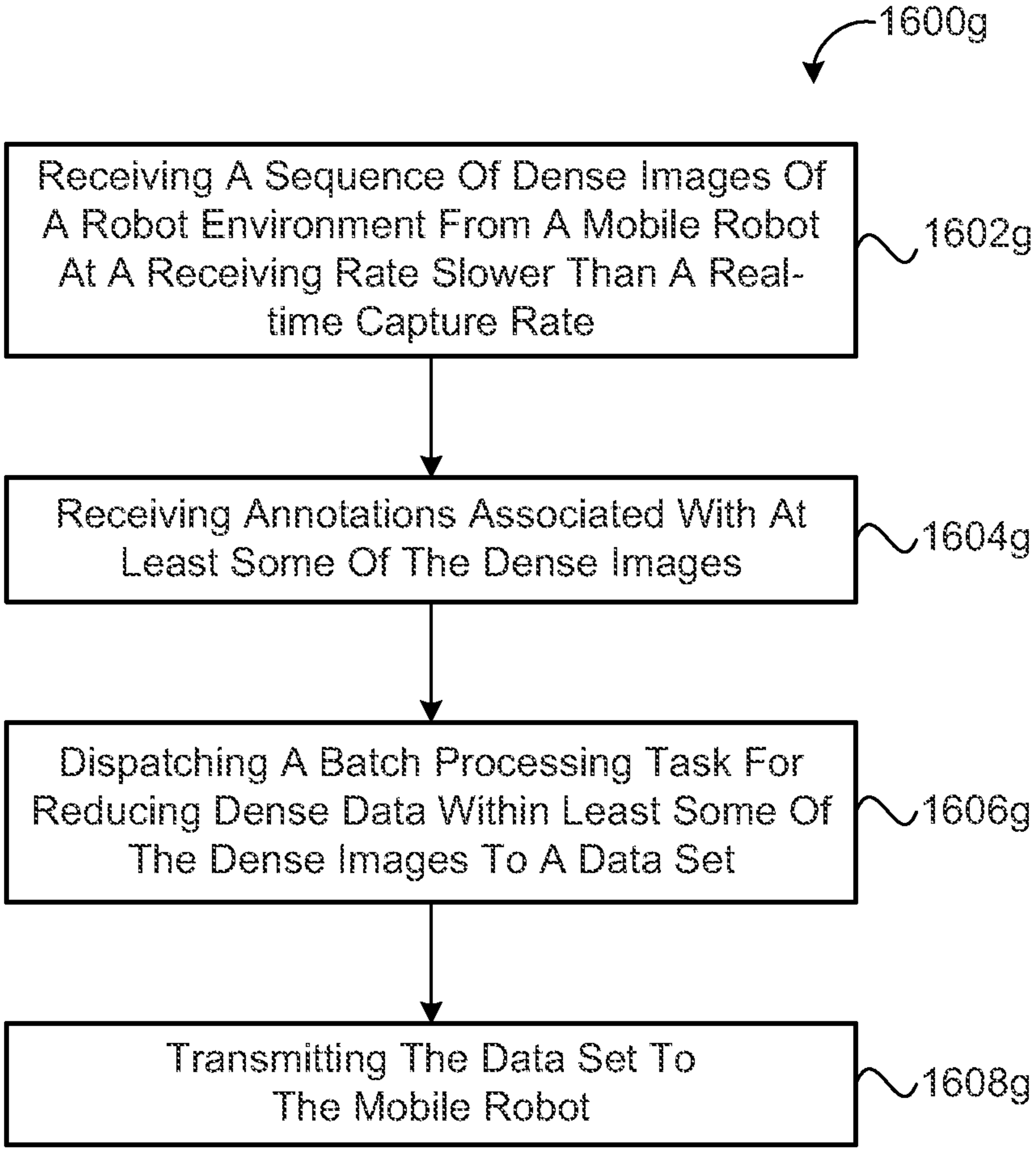
1600g
Receiving A Sequence Of Dense Images Of A Robot Environment From A Mobile Robot At A Receiving Rate Slower Than A Real-time Capture Rate
1602g
Receiving Annotations Associated With At Least Some Of The Dense Images
1604g
Dispatching A Batch Processing Task For Reducing Dense Data Within Least Some Of The Dense Images To A Data Set
1606g
Transmitting The Data Set To The Mobile Robot
1608g

FIG. 16G

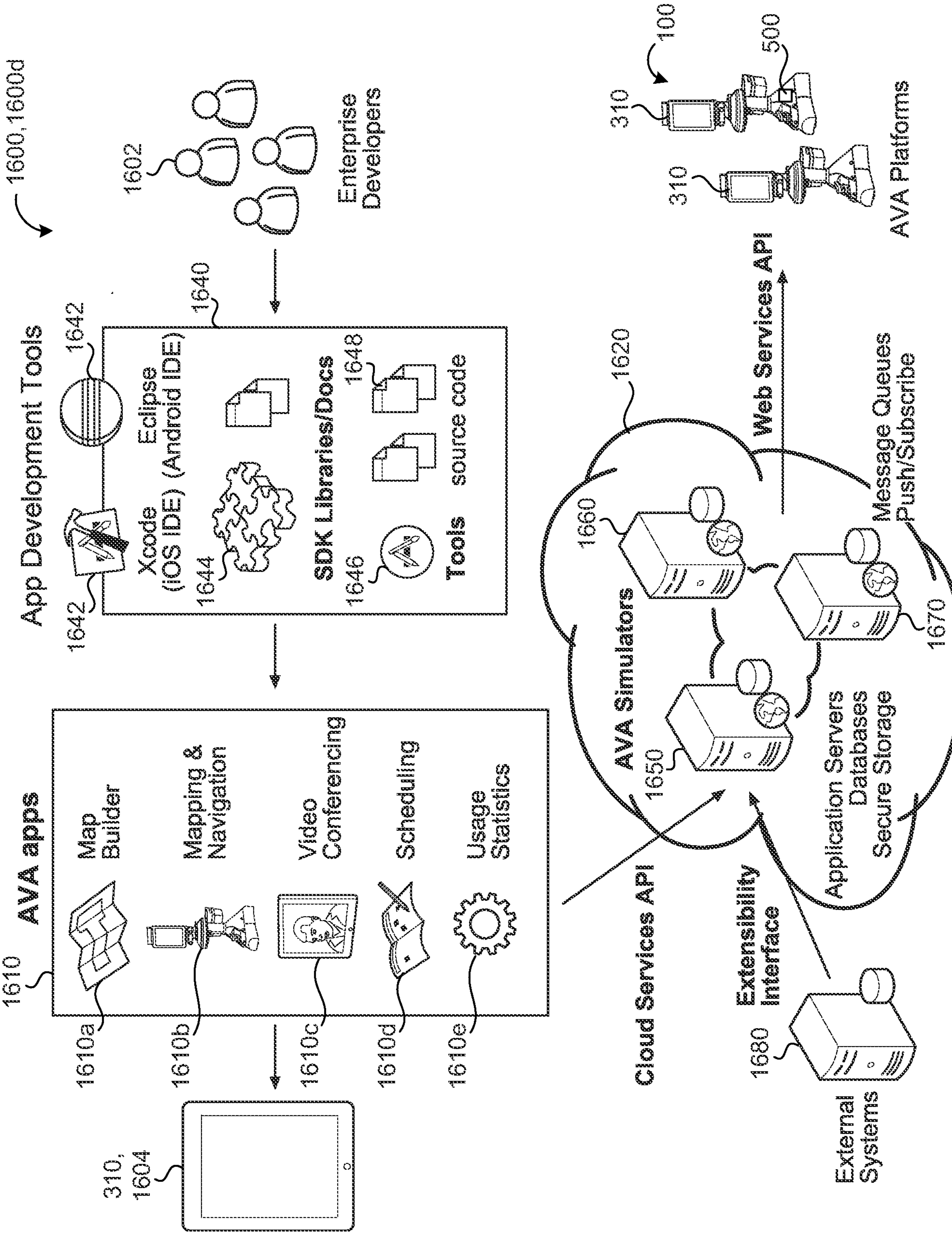
1600,1600d
1602
Enterprise Developers
App Development Tools
1640
1642
Xcode (iOS IDE)
1642
Eclipse (Android IDE)
1644
SDK Libraries/Docs
1646
Tools
1648
source code
source code
AVA apps
1610
Map Builder
Mapping & Navigation
Video Conferencing
Scheduling
Usage Statistics
1610a
1610b
1610c
1610d
1610e
310, 1604
1620
AVA Simulators
1650
1660
1670
Application Servers
Databases
Secure Storage
Message Queues
Push/Subscribe
Web Services API
Cloud Services API
Extensibility Interface
1680
External Systems
100
500
310
310
AVA Platforms

FIG. 16H

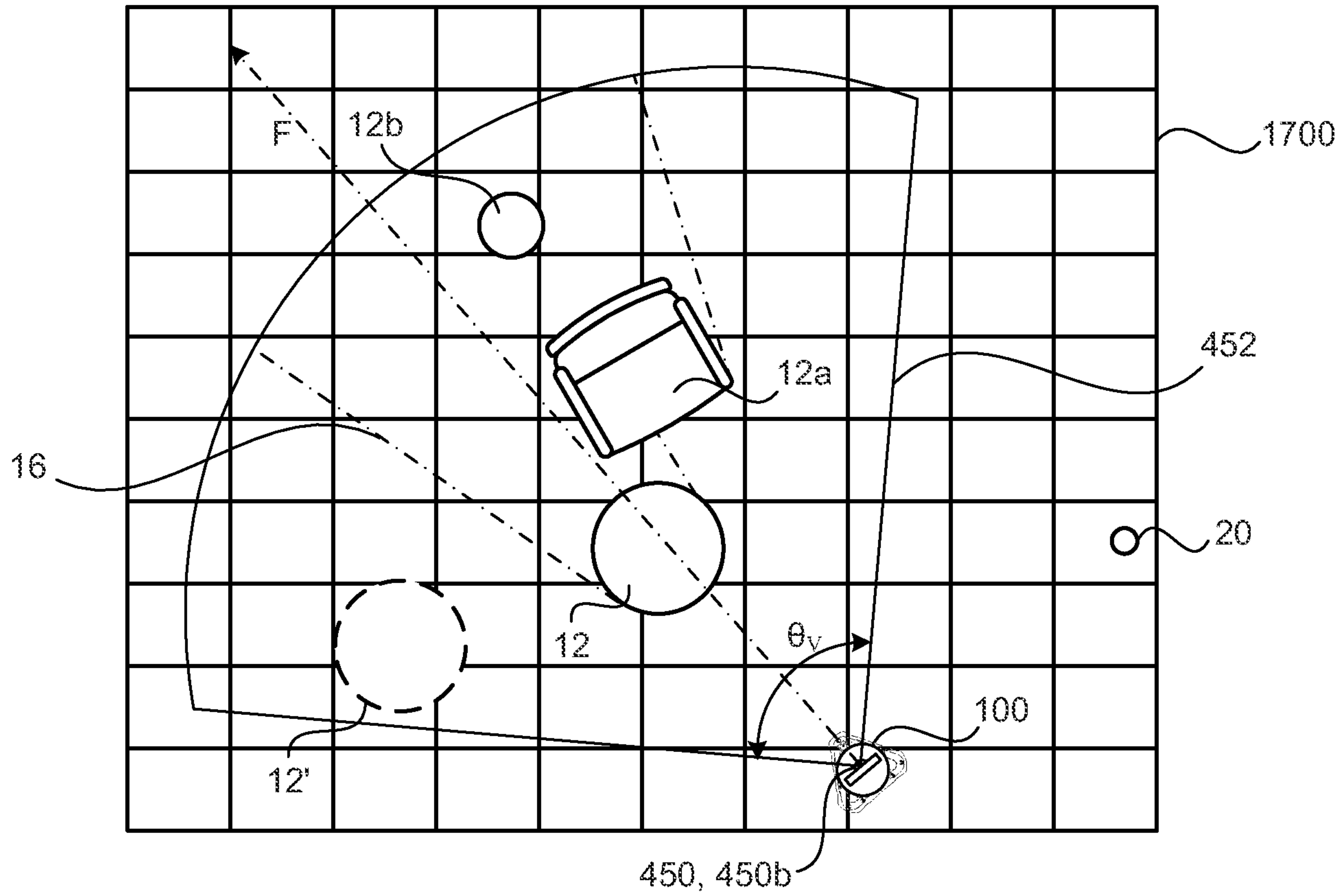

FIG. 17A

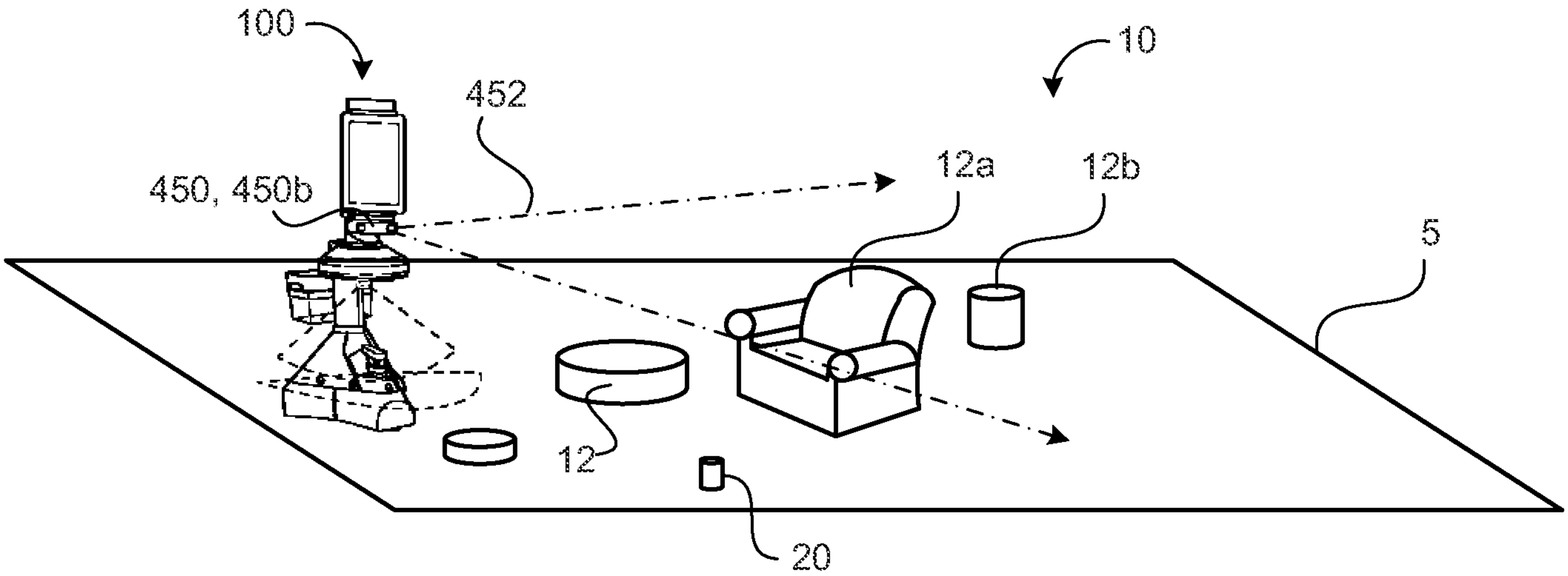

FIG. 17B

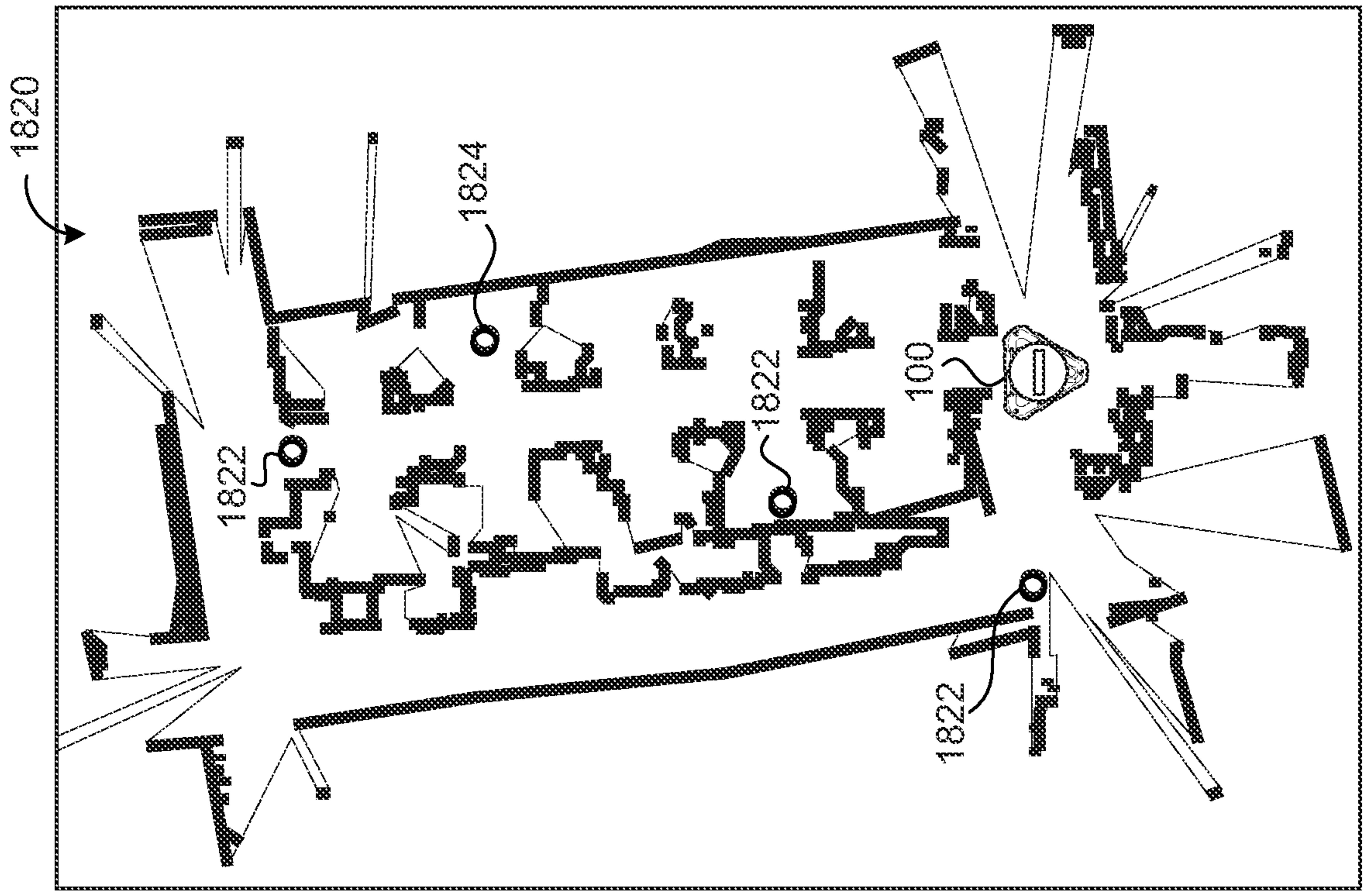

FIG. 18B

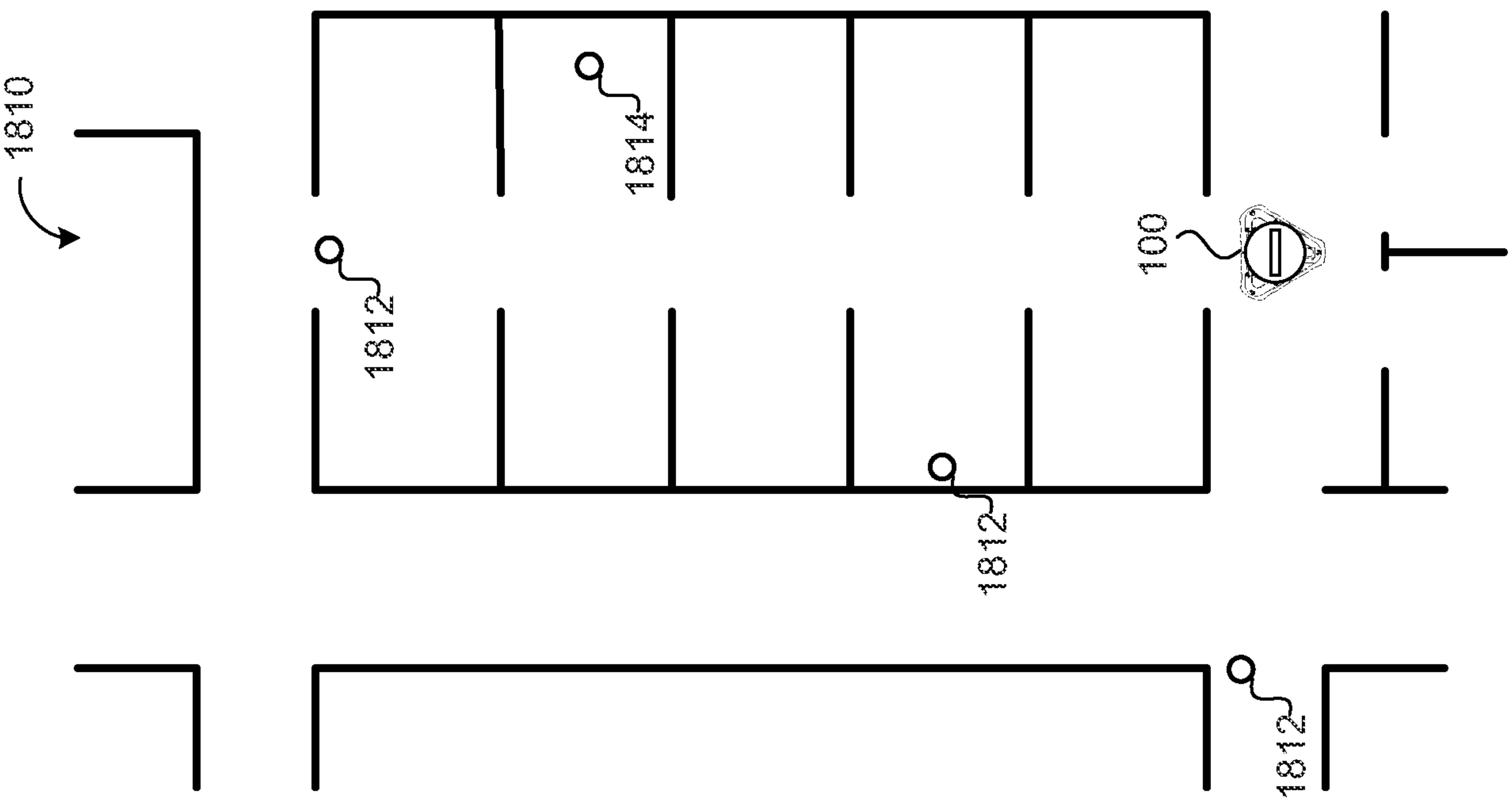

FIG. 18A

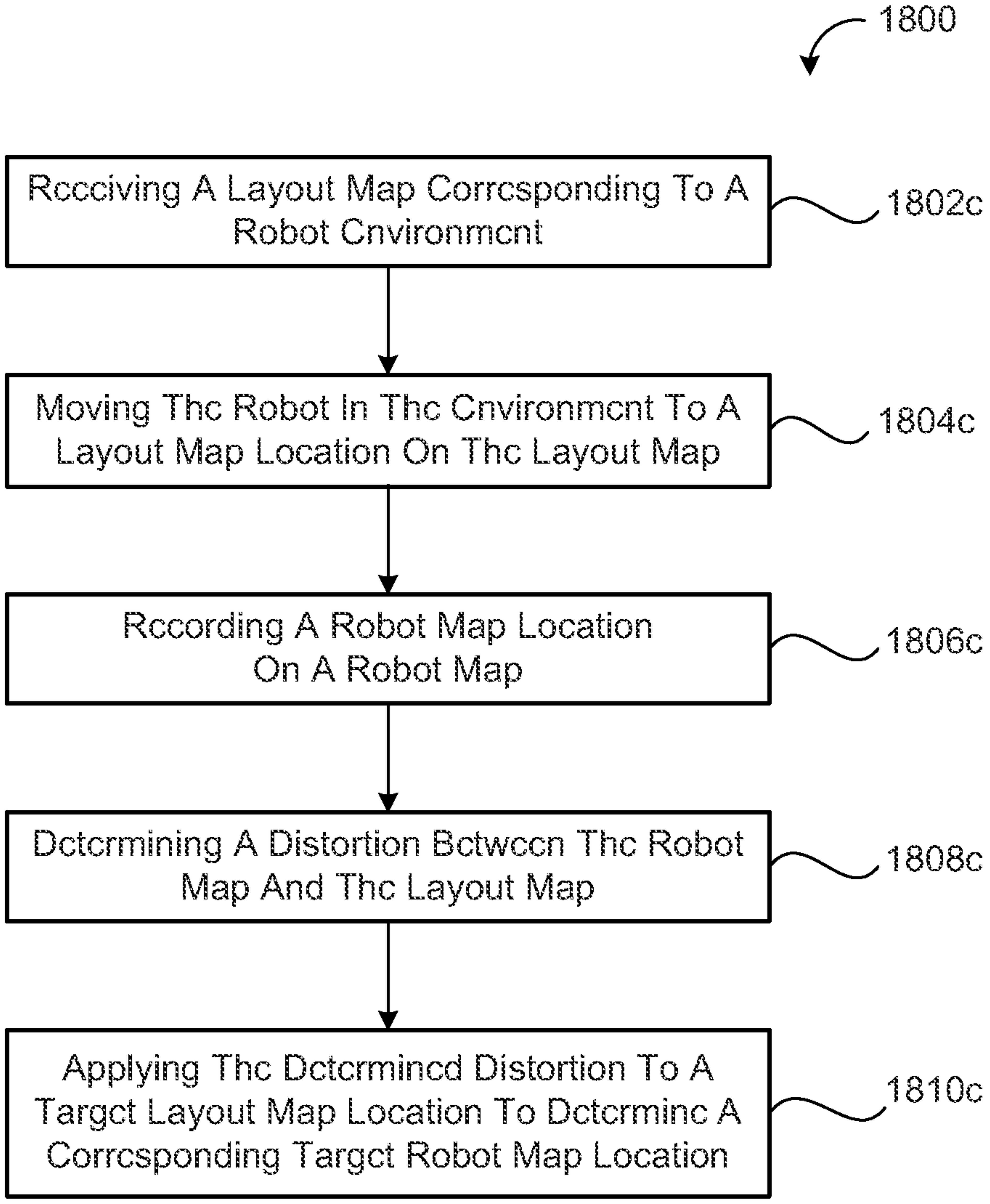
1800
Rccciving A Layout Map Corrcsponding To A Robot Cnvironmcnt
1802c
Moving Thc Robot In Thc Cnvironmcnt To A Layout Map Location On Thc Layout Map
1804c
Rccording A Robot Map Location On A Robot Map
1806c
Dctcrmining A Distortion Bctwccn Thc Robot Map And Thc Layout Map
1808c
Applying Thc Dctcrmincd Distortion To A Targct Layout Map Location To Dctcrminc A Corrcsponding Targct Robot Map Location
1810c

FIG. 18C

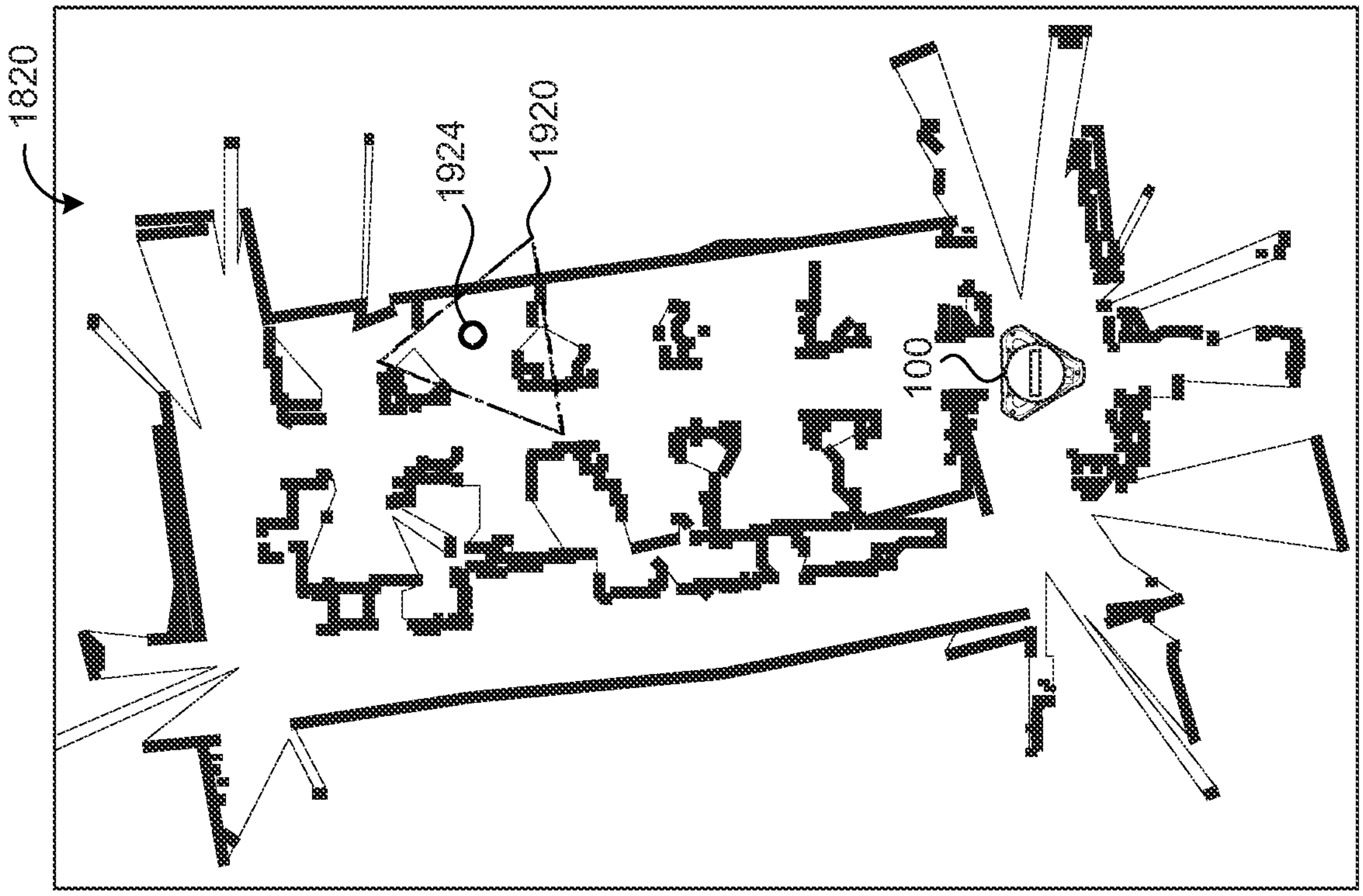

FIG. 19B

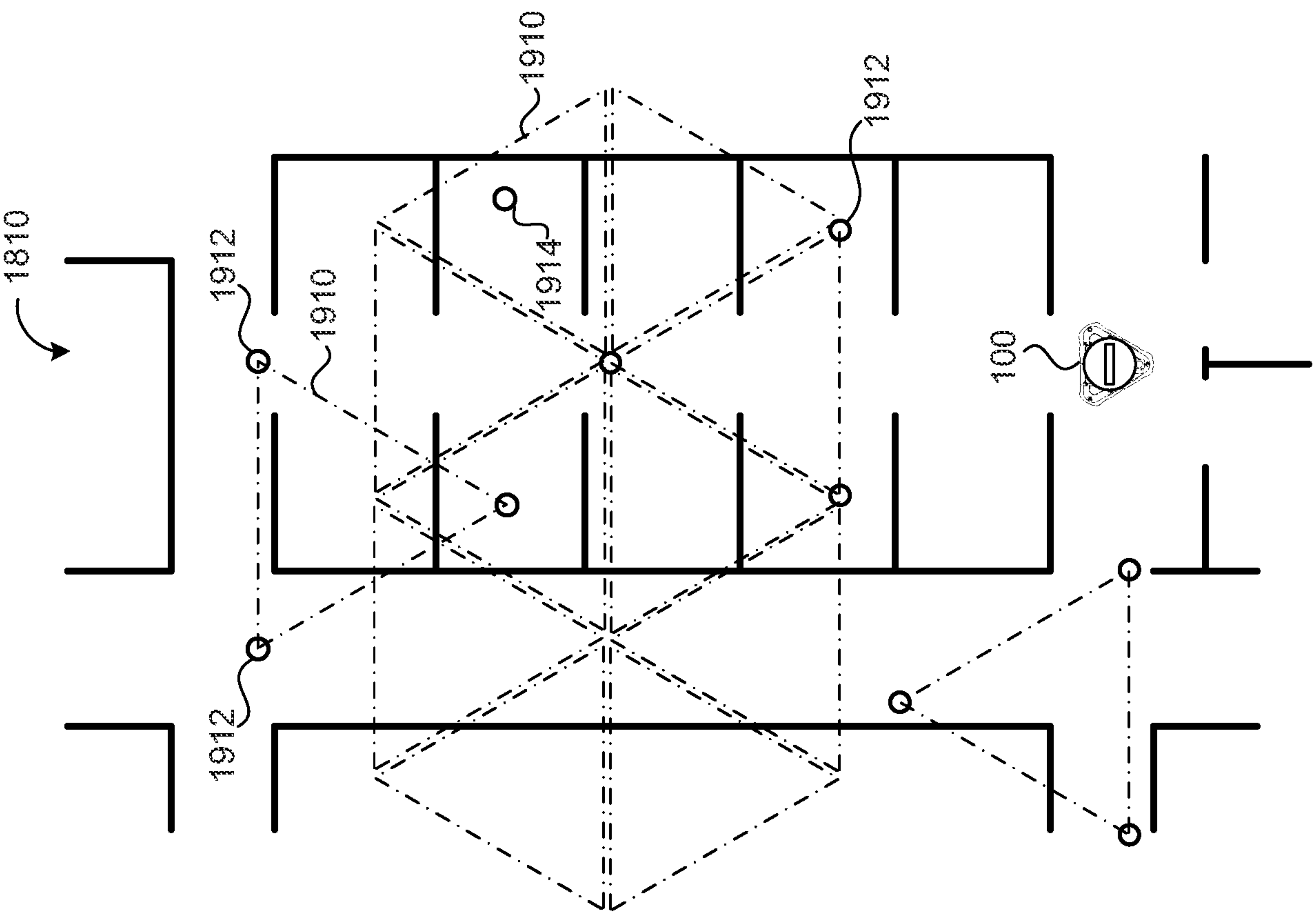

FIG. 19A

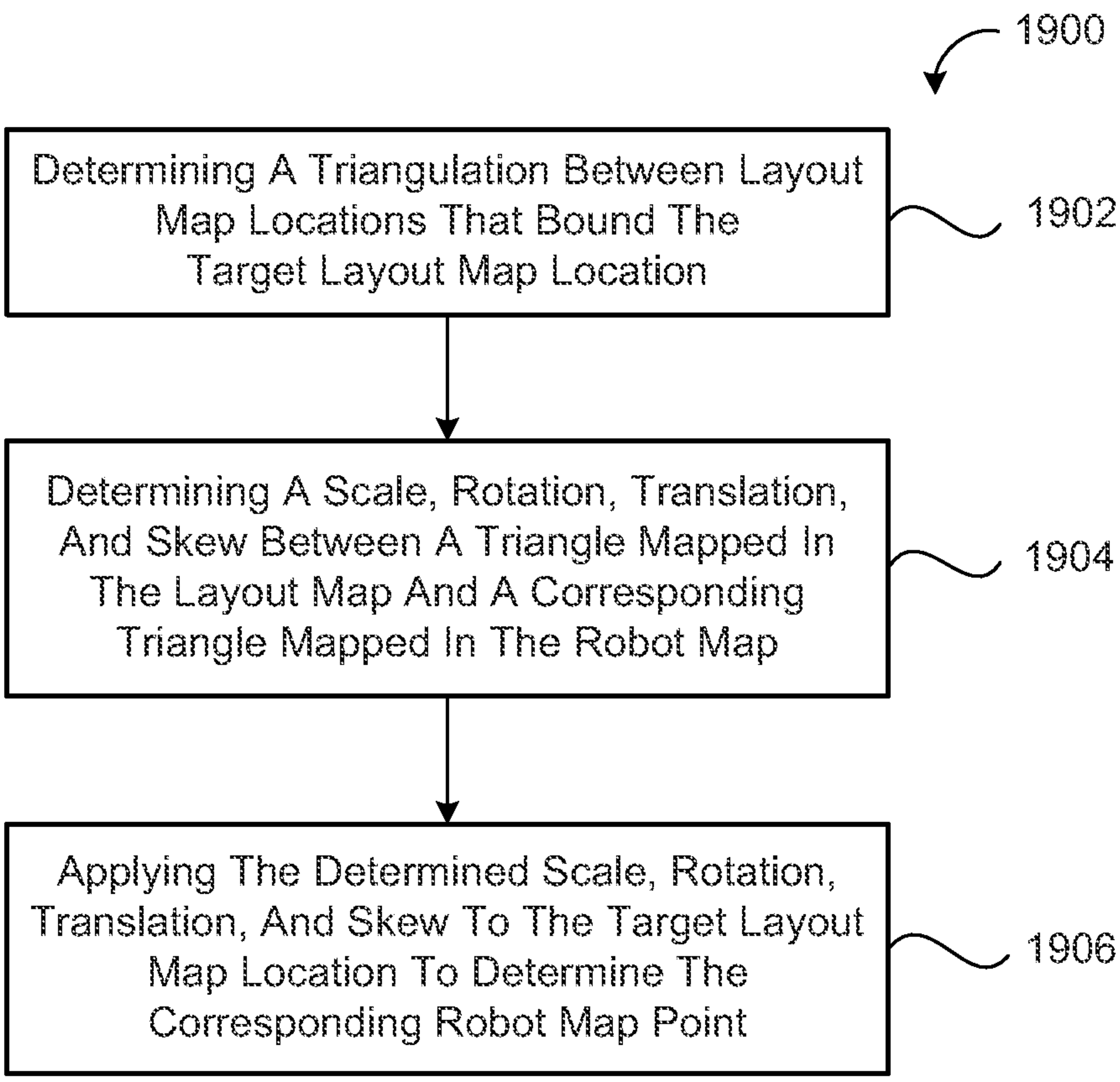
1900
Determining A Triangulation Between Layout Map Locations That Bound The Target Layout Map Location
1902
Determining A Scale, Rotation, Translation, And Skew Between A Triangle Mapped In The Layout Map And A Corresponding Triangle Mapped In The Robot Map
1904
Applying The Determined Scale, Rotation, Translation, And Skew To The Target Layout Map Location To Determine The Corresponding Robot Map Point
1906

FIG. 19C

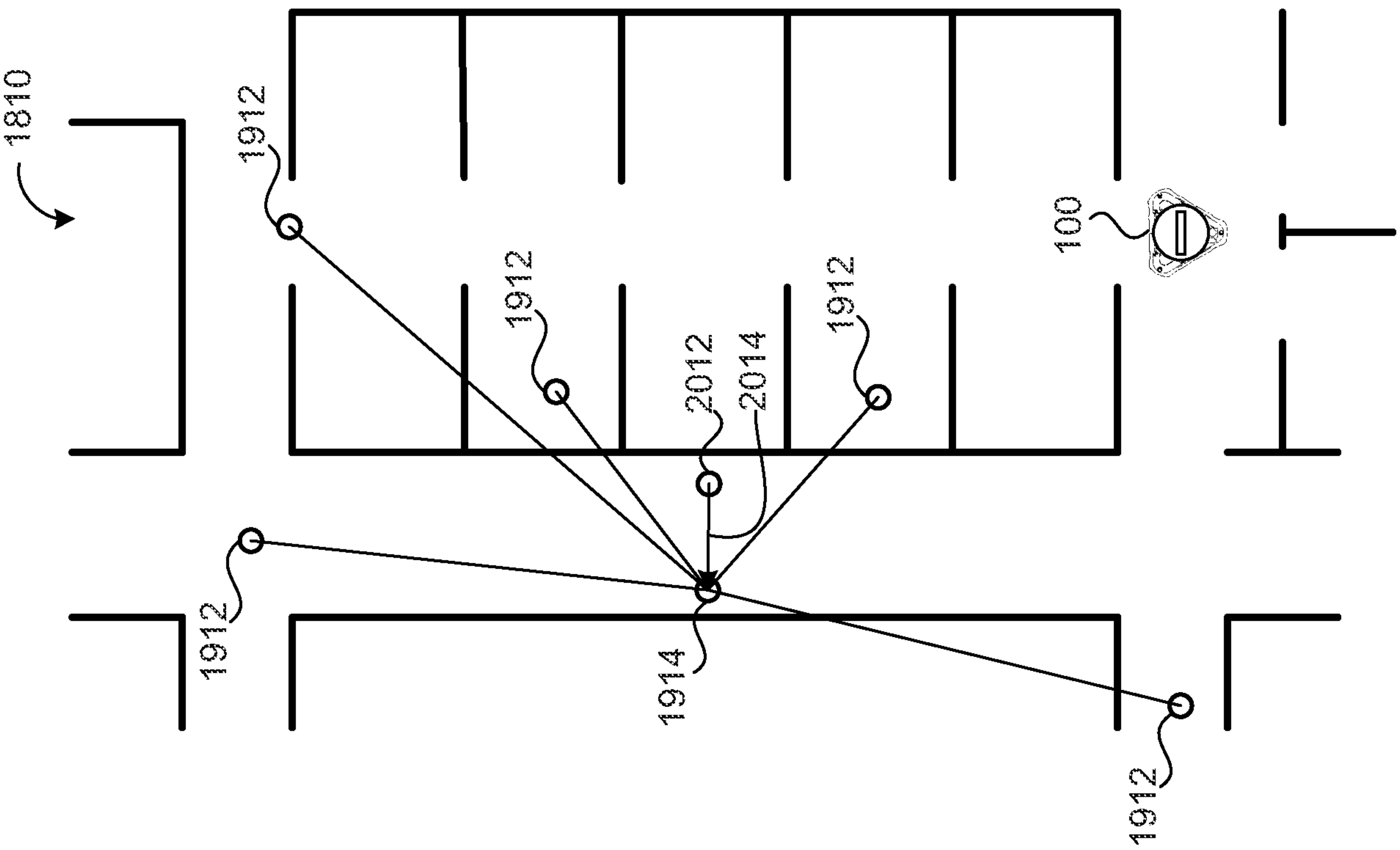

FIG. 20A

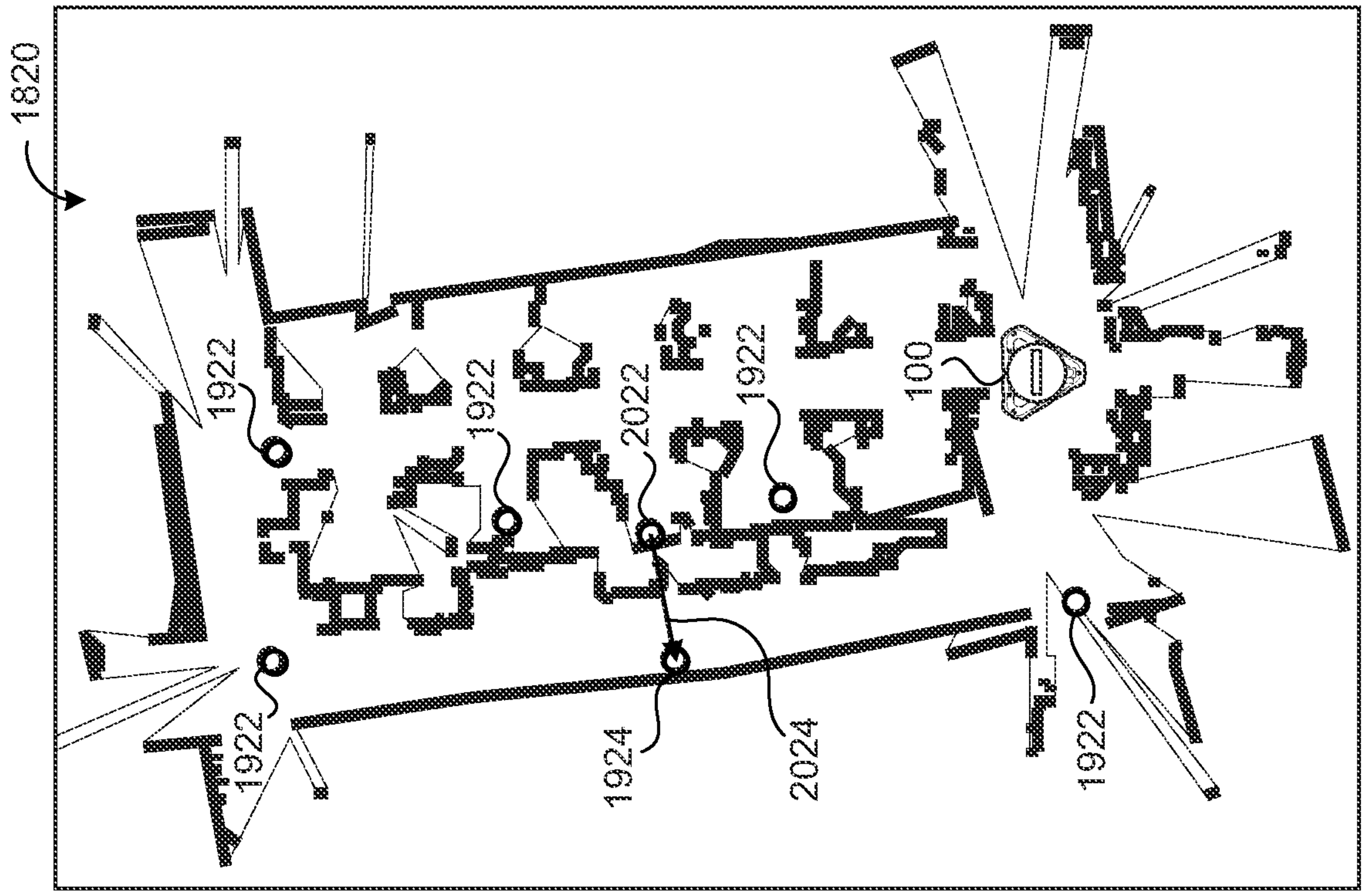

FIG. 20B

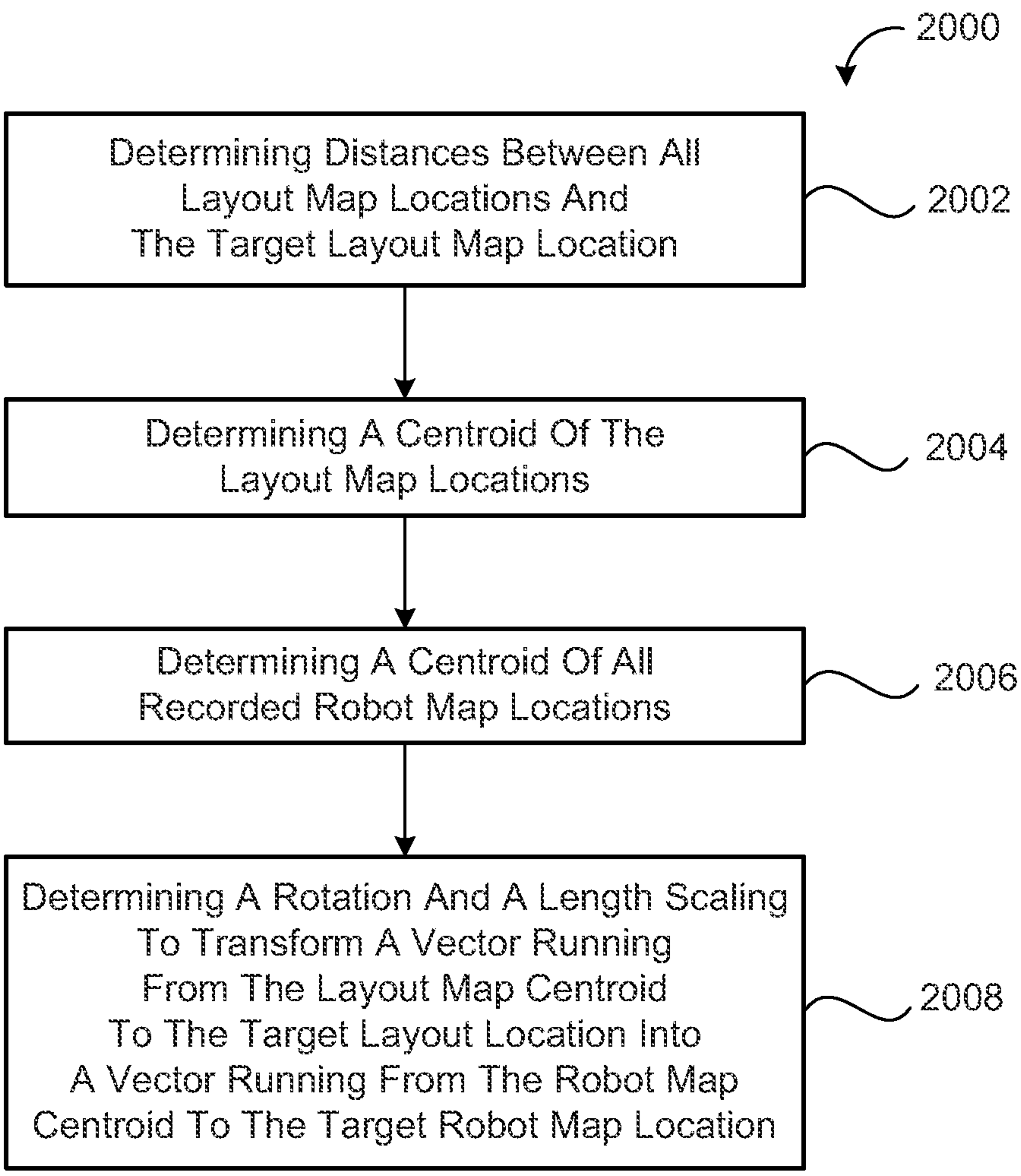
2000
Determining Distances Between All Layout Map Locations And The Target Layout Map Location
2002
Determining A Centroid Of The Layout Map Locations
2004
Determining A Centroid Of All Recorded Robot Map Locations
2006
Determining A Rotation And A Length Scaling To Transform A Vector Running From The Layout Map Centroid To The Target Layout Location Into A Vector Running From The Robot Map Centroid To The Target Robot Map Location
2008

FIG. 20C

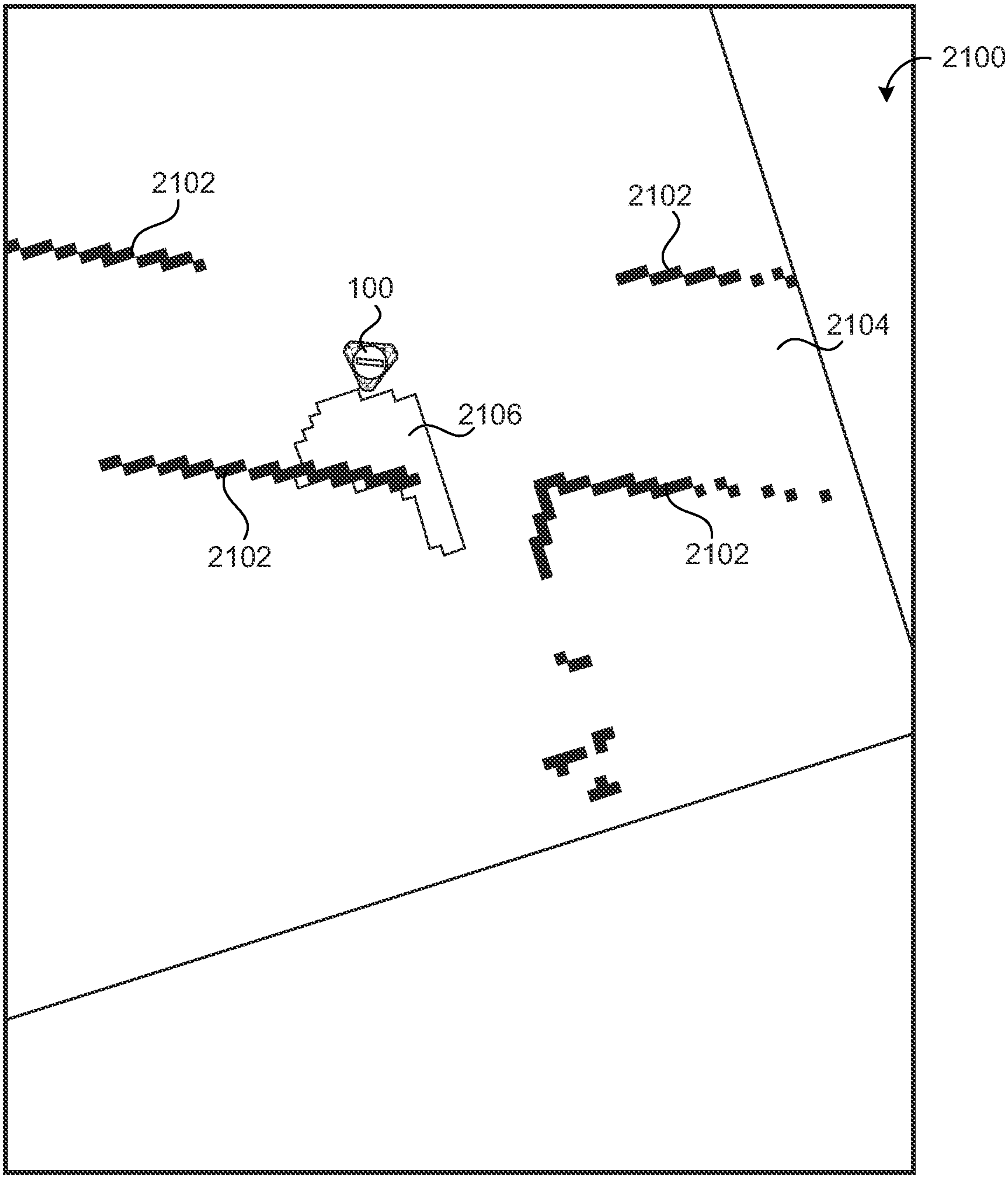

FIG. 21A

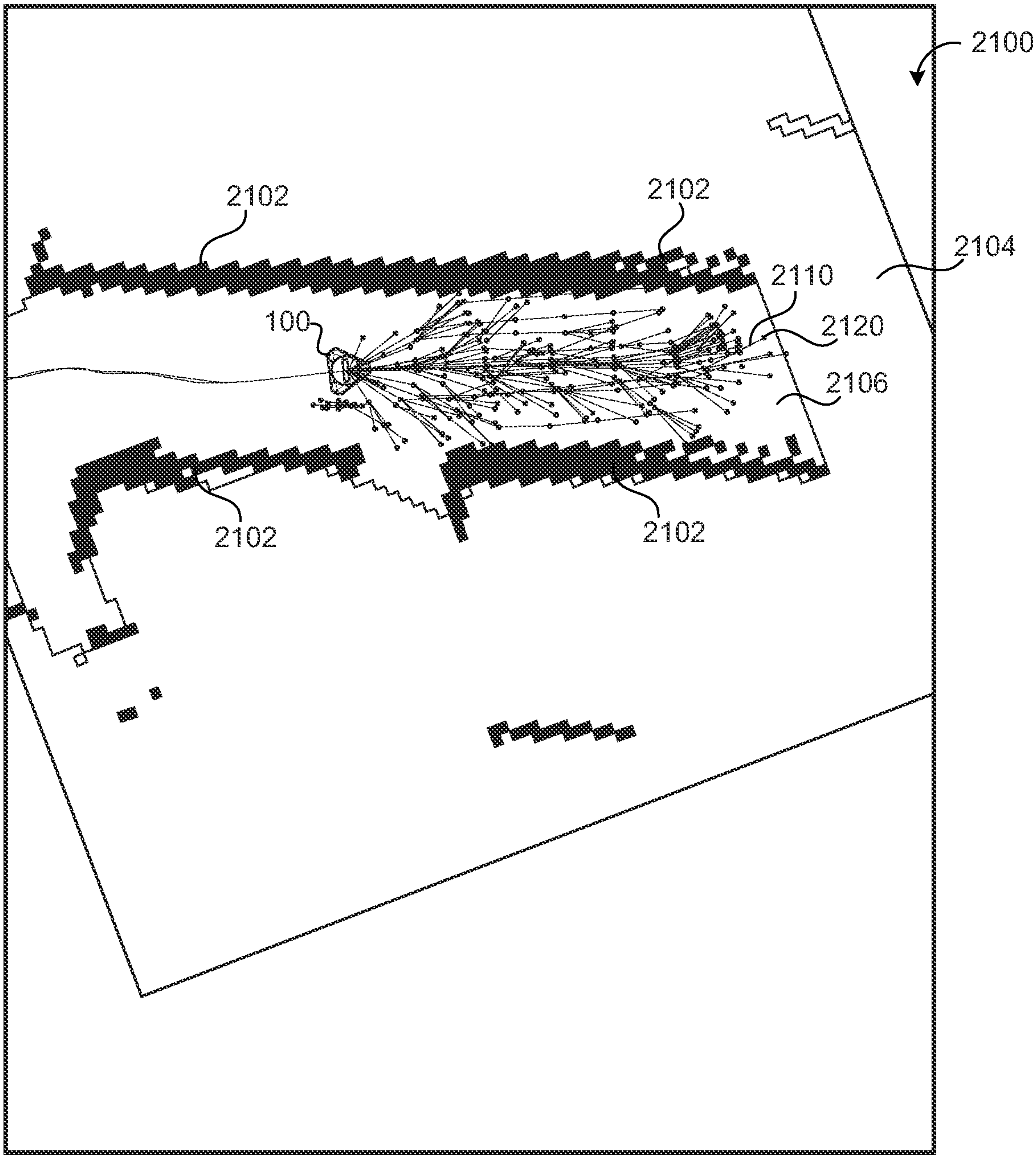

FIG. 21B

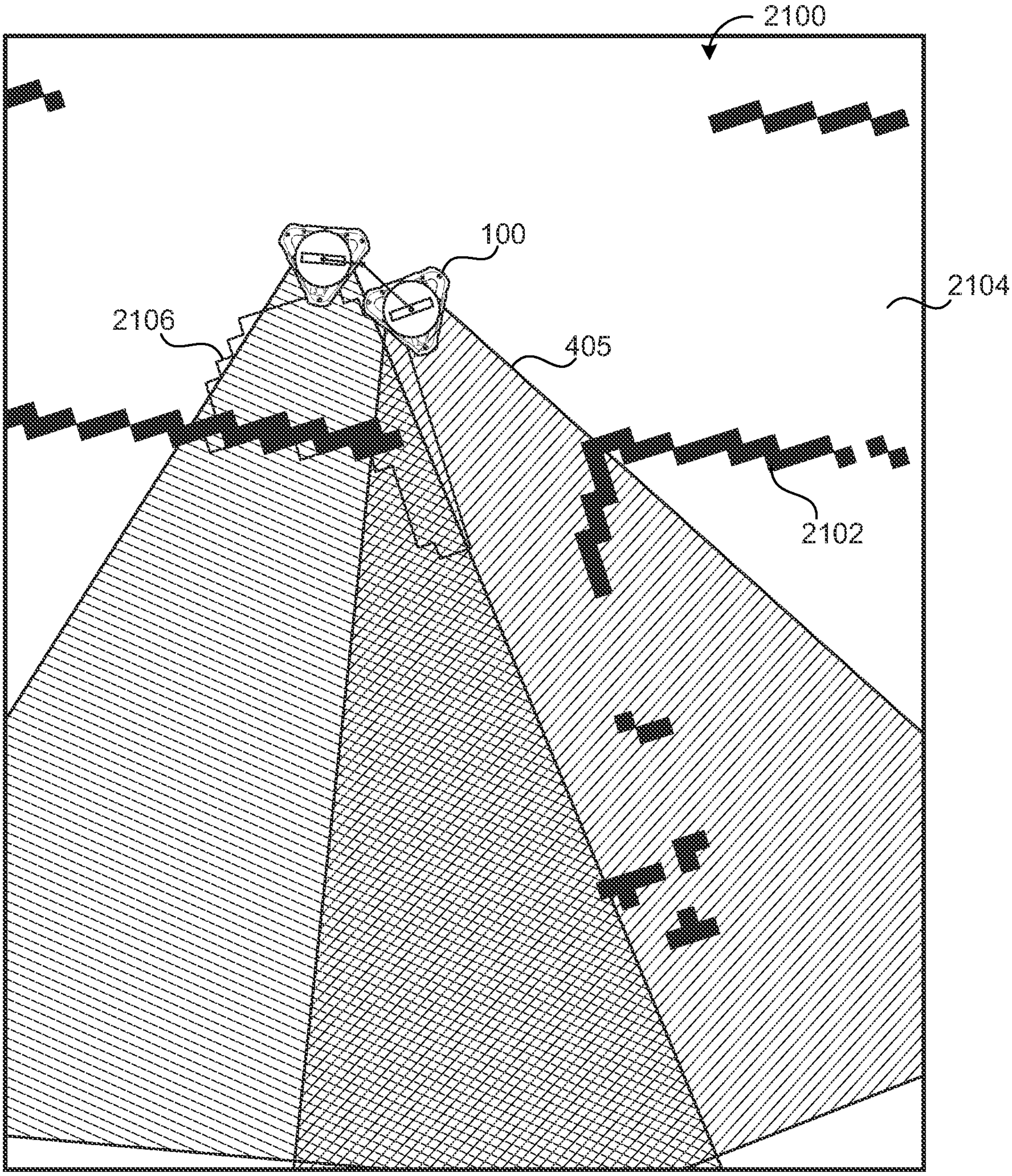

FIG. 21C

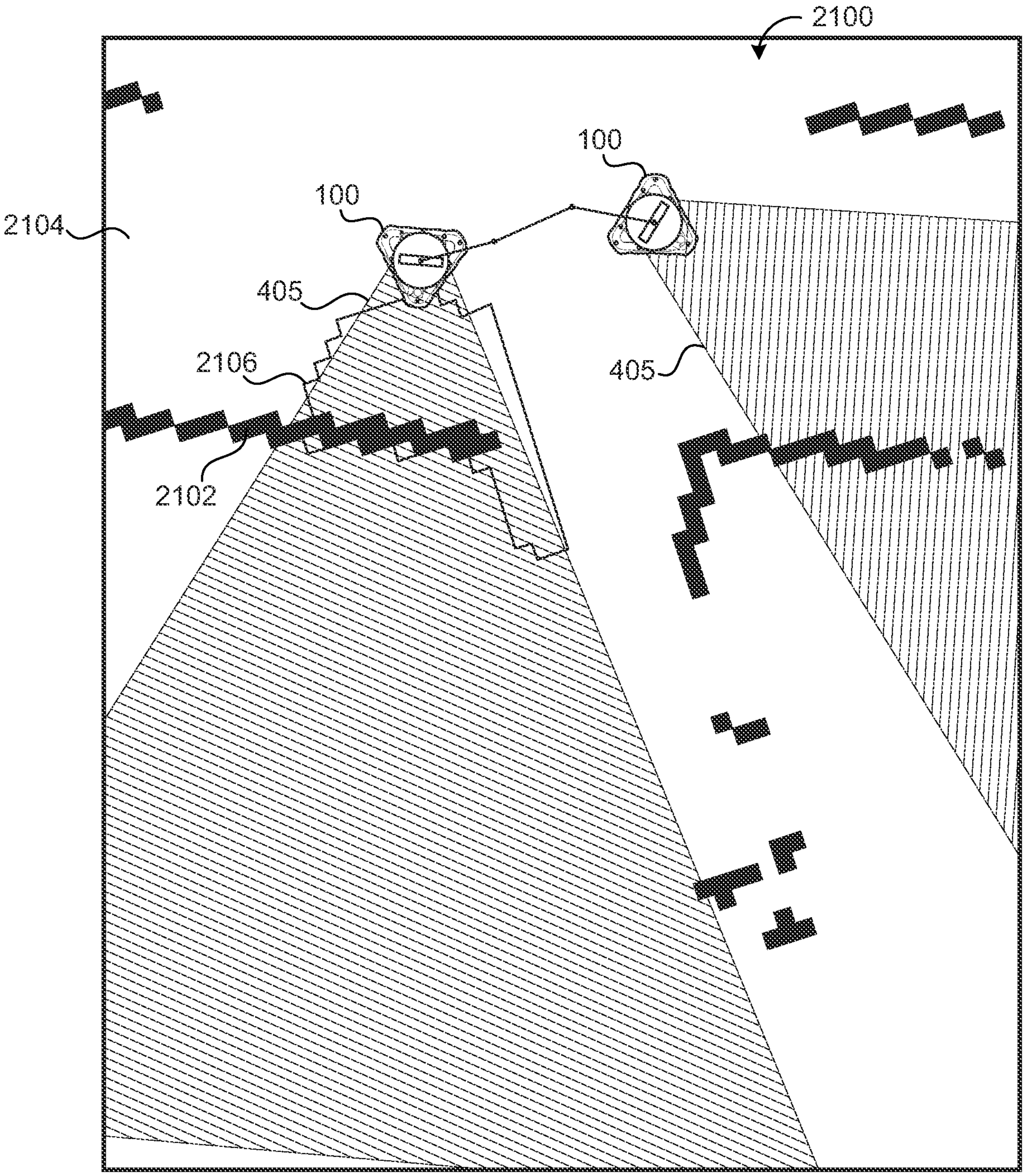

FIG. 21D

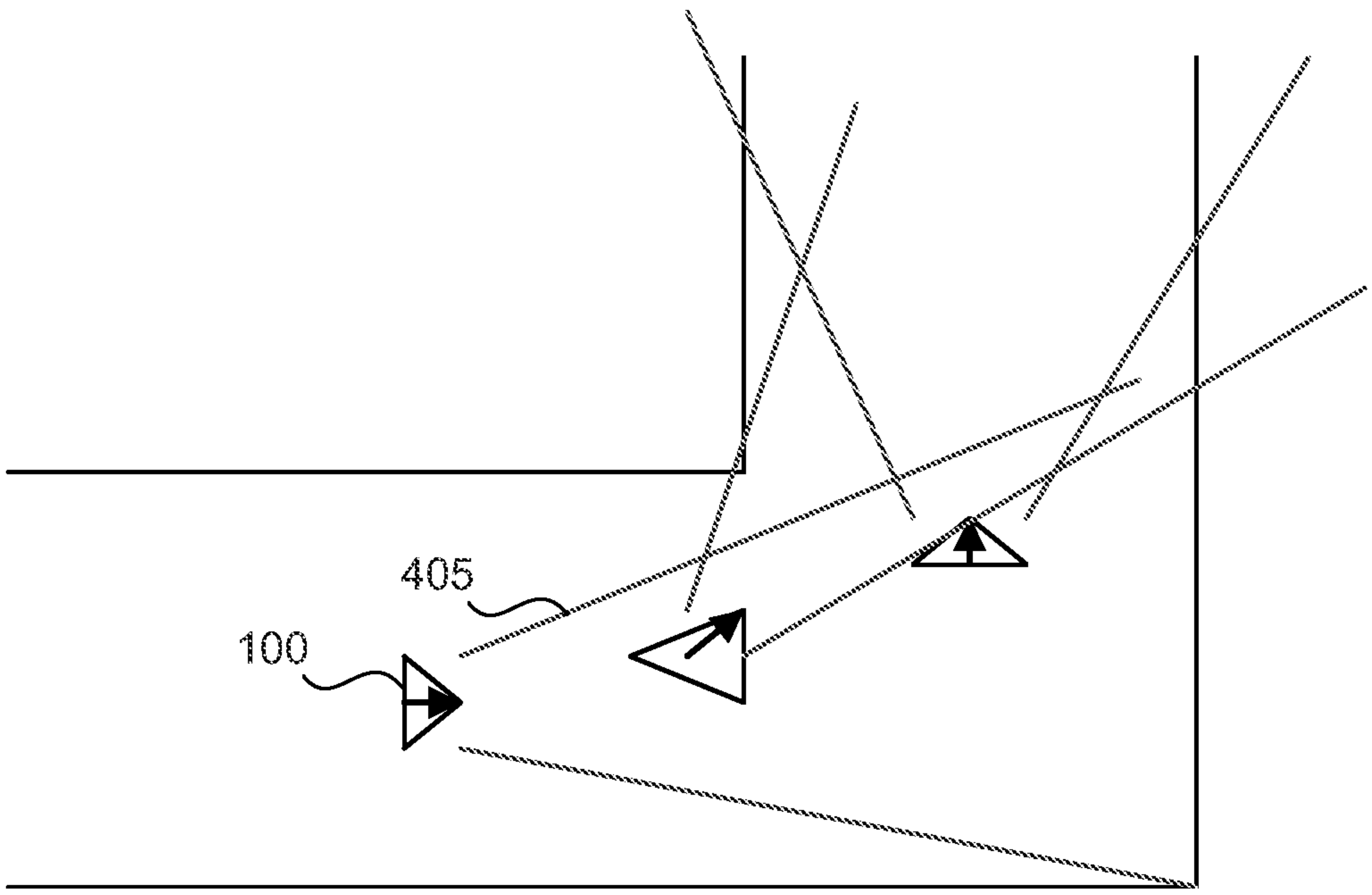

FIG. 21E

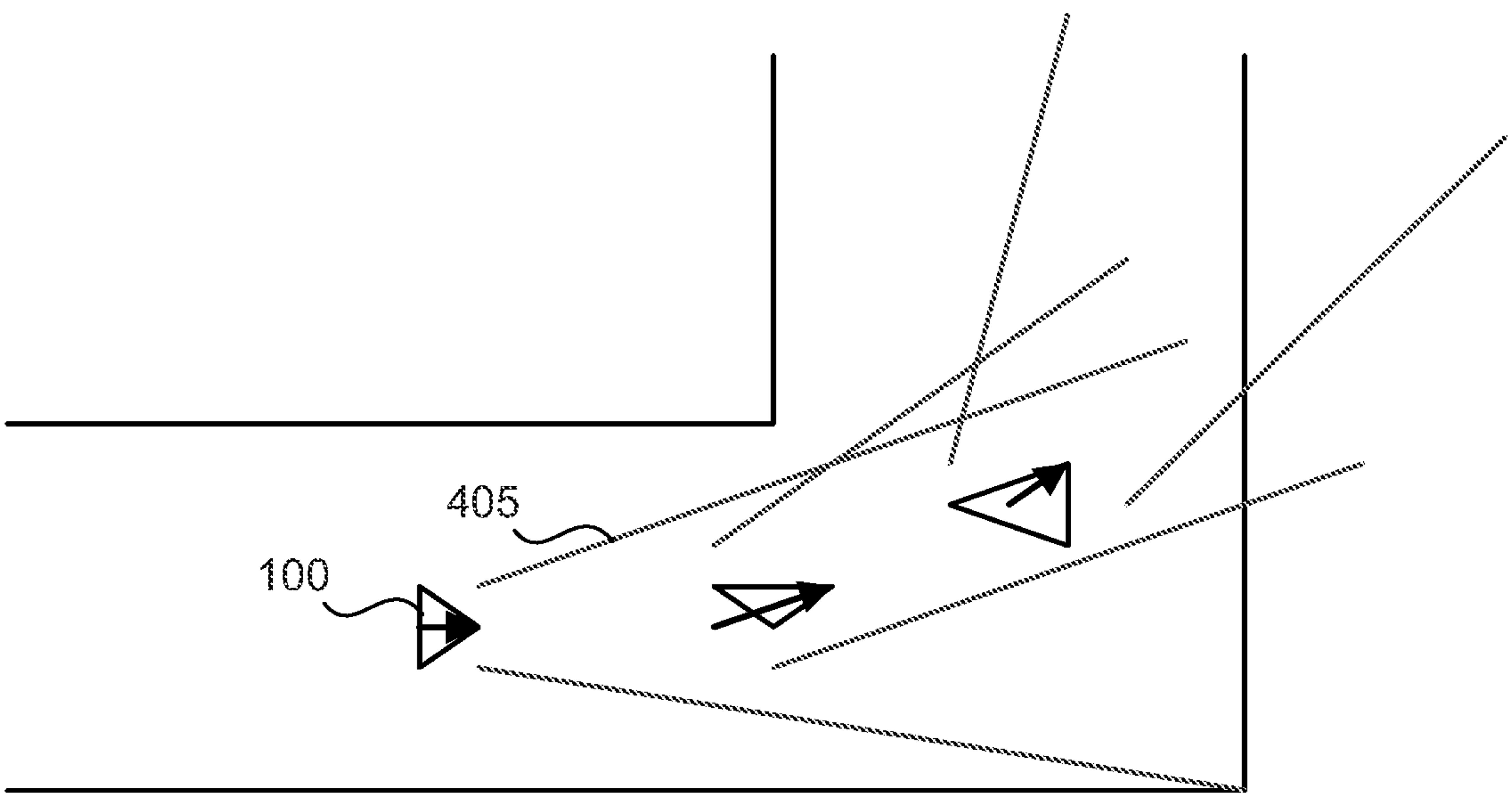

FIG. 21F

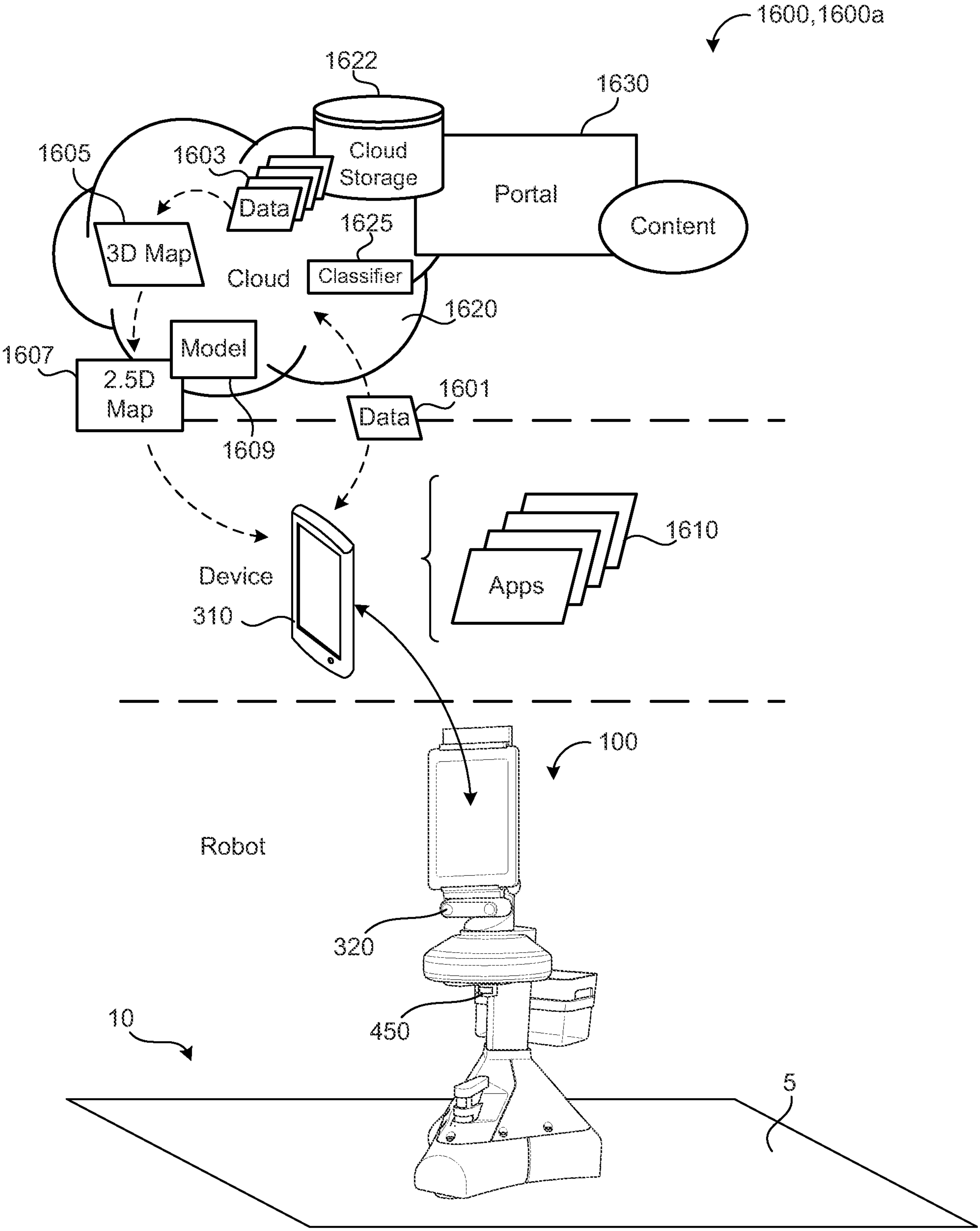
1600,1600a
1622
1630
Cloud
Storage
1605
1603
Portal
Data
Content
1625
3D Map
Cloud
Classifier
1620
1607
Model
2.5D
Map
Data
1601
1609
1610
Apps
Device
310
100
Robot
320
450
10
5

FIG. 16A