



(51) International Patent Classification:

A63F 13/211 (2014.01) G06F 3/03 (2006.01)
 A63F 13/218 (2014.01) G06T 19/20 (2011.01)
 A63F 13/428 (2014.01)

(21) International Application Number:

PCT/US2019/037794

(22) International Filing Date:

18 June 2019 (18.06.2019)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

62/687,774	20 June 2018 (20.06.2018)	US
62/687,806	21 June 2018 (21.06.2018)	US
16/389,499	19 April 2019 (19.04.2019)	US

(71) Applicant: **VALVE CORPORATION** [US/US]; 10400 NE 4th Street, Bellevue, WA 98004 (US).

(72) Inventors: **BRADNER, Keith**; c/o Valve Corporation, 10400 NE 4th Street, Bellevue, WA 98004 (US). **SLOCUM, Jeremy**; c/o Valve Corporation, 10400 NE 4th Street, Bellevue, WA 98004 (US). **NIETFELD, Scott Douglas**; c/o Valve Corporation, 10400 NE 4th Street, Bellevue, WA 98004 (US). **YANG, Lawrence**; c/o Valve Corporation, 10400 NE 4th Street, Bellevue, WA 98004 (US).

poration, 10400 NE 4th Street, Bellevue, WA 98004 (US). **LEINBAUGH, Jeffrey George**; c/o Valve Corporation, 10400 NE 4th Street, Bellevue, WA 98004 (US).

(74) Agent: **HARTMAN, Robert G.**; Lee & Hayes, P.C., 601 W. Riverside Ave, Suite 1400, Spokane, WA 99201 (US).

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

(54) Title: HOLDING AND RELEASING VIRTUAL OBJECTS

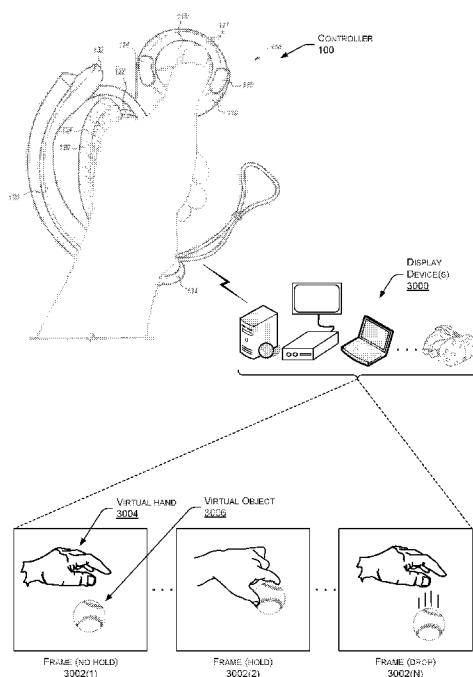


FIG. 30

(57) Abstract: Techniques and devices for holding and releasing virtual objects on a display based on input received from one or more handheld controllers are described herein. In some instances, a handheld controller includes one or more sensors, such as proximity sensors, force sensors (e.g., force resisting sensors, etc.), accelerometers, and/or other types of sensors configured to receive input from a hand of a user gripping the handheld controller. Hardware, software, and/or firmware on the controller and/or on a device coupled to the controller (e.g., a game console, a server, etc.) may receive data from these sensors and generate a representation of a corresponding gesture on a display, such as a monitor, a virtual-reality system, and/or the like.

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW,
KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

HOLDING AND RELEASING VIRTUAL OBJECTS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Patent Application No. 16/389,499 filed April 19, 2019, which
5 claims priority under 35 U.S.C. §120 as a continuation to pending U.S. Provisional Patent Application Serial No. 62/687,774, filed on June 20, 2018 and entitled HOLDING AND RELEASING VIRTUAL OBJECTS, which also claims priority under 35 U.S.C. §120 as a continuation to pending U.S. Provisional Patent Application Serial No. 62/687,806, filed on June 21, 2018 and entitled STEAMVR KNUCKLES DRIVER.

[0002] Further, this application claims priority under 35 U.S.C. §120 as a continuation-in-part to pending
10 U.S. Patent Application Serial No. 15/834,372 filed on 7 December 2017, entitled “ELECTRONIC CONTROLLER WITH FINGER SENSING AND AN ADJUSTABLE HAND RETAINER,” which itself claims priority as a continuation-in-part to U.S. Patent Application Serial No. 15/679,521 filed on 17 August 2017, entitled “ELECTRONIC CONTROLLER WITH HAND RETAINER AND FINGER MOTION SENSING,” which itself claims priority as a continuation-in-part to U.S. Patent Application Serial No. 29/580,635 filed 11
15 October 2016, and claims priority to U.S. Provisional Patent Application 62/520,958 filed 16 June 2017.

BACKGROUND OF THE DISCLOSURE

[0003] The video game industry has become large and important, and has spawned many innovations in both software and related hardware. Various hand-held video game controllers have been designed, manufactured, and sold, for a variety of game applications. Some of those innovations have applicability outside of the video game
20 industry, such as for controllers of industrial machines, defense systems, robotics, etc. Virtual reality (VR) systems are an application of great contemporary interest and rapid technical advancement, both within and outside of the video game industry. The controllers for VR systems have to perform several different functions, and meet strict (and sometimes competing) design constraints, often while optimizing certain desired characteristics like ease of use, etc.

[0004] One example objective of controllers used in VR systems is to mimic natural interactions, such as grasping, throwing, squeezing, or otherwise interacting with virtual objects. However, VR and other types of systems may have difficulty determining when a user intends to grasp, throw, squeeze, or otherwise manipulate virtual objects and, thus, have difficulty accurately depicting these interactions on a display.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] Fig. 1 depicts a controller according to an example embodiment of the present disclosure, with a hand
30 retainer in an open position.

[0006] Fig. 2 depicts the controller of Fig. 1 in an open, palm-up hand of a user.

[0007] Fig. 3 depicts the controller of Fig. 1 in a closed hand of the user.

[0008] Fig. 4 depicts the controller of Fig. 1 in a palm-down hand of the user.

[0009] Fig. 5 depicts a pair of controllers according to an example embodiment of the present disclosure,
35 with hand retainers in an open position.

[00010] Fig. 6A depicts a front view of right-hand controller according to another example embodiment of the present disclosure.

[00011] Fig. 6B depicts a back view of the right-hand controller of Fig. 6A.

[0012] Fig. 7A depicts a window for an infrared light sensor, according to an embodiment of the present disclosure.

[0013] Fig. 7B depicts a window for an infrared light sensor, according to another embodiment of the present disclosure.

5 [0014] Fig. 8 shows a side view of the right-hand controller of Fig. 6A, with an outer shell that partially wraps the tubular housing of the controller's handle being exploded away to reveal instrumentation on its inner surface.

[0015] Fig. 9A depicts a cross section of the right-hand controller of Fig. 6A, with an outer shell that partially wraps the tubular housing of the controller's handle being exploded away.

10 [0016] Fig. 9B depicts the cross section of Fig. 9A, except with the outer shell installed in its normal operational position.

[0017] Fig. 10A depicts a front view of right-hand controller according to another example embodiment of the present disclosure, with a partially-closed hand retainer.

[0018] Fig. 10B depicts a front view the controller of Fig. 10A, except with the hand retainer fully open.

15 [0019] Fig. 11A depicts a front view of head and handle components of a controller according to an example embodiment of the present disclosure, including a hand retainer anchor that can move peripherally about the head.

[0020] Fig. 11B depicts the head and handle components of Fig. 11A except with a faceplate removed from the head to expose a lockable collar portion that may facilitate selective adjustment of the hand retainer anchor peripherally about the head.

20 [0021] Fig. 12A depicts a partially assembled controller according to an alternative embodiment of the present disclosure, with a hand retainer component removed.

[0022] Fig. 12B depicts a closer view of a channel feature of the controller of Fig. 12A.

[0023] Fig. 12C is a cross-sectional view of the channel depicted in Fig. 12B.

25 [0024] Fig. 13A depicts a force sensing resistor (FSR) according to an example embodiment of the present disclosure.

[0025] Fig. 13B depicts a front view of the FSR of Fig. 13A.

[0026] Fig. 13C depicts a cross section of the FSR of Fig. 13B, taken along Section A-A, the cross section showing the first substrate made of polyimide.

30 [0027] Fig. 14 depicts various front views of a FSR at progressive stages in an example process of constructing the FSR.

[0028] Fig. 15 depicts example layers of a FSR, according to another embodiment of the present disclosure. Fig. 15 is not to scale. Rather, Fig. 15 is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR.

35 [0029] Fig. 16 depicts example layers of a FSR, according to another embodiment of the present disclosure. Fig. 16 is not to scale. Rather, Fig. 16 is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR.

[0030] Fig. 17 depicts example layers of a FSR, according to another embodiment of the present disclosure. Fig. 17 is not to scale. Rather, Fig. 17 is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR.

40

[0031] Fig. 18A depicts a front view of a FSR, before a folding step to form the complete FSR, according to another embodiment of the present disclosure.

[0032] Fig. 18B depicts a front view of the FSR of Fig. 18A, after the folding step is performed.

[0033] Fig. 18C depicts a cross section of the FSR of Fig. 18A, taken along Section B-B.

5 [0034] Fig. 18D depicts example layers of the FSR of Fig. 18A. Fig. 18D is not to scale. Rather, Fig. 18D is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR.

[0035] Fig. 19 is a flow diagram of an example process for manufacturing a FSR.

10 [0036] Fig. 20 illustrates an example user interface (UI) that may be used to configure a FSR-based input mechanism of a controller for an electronic system to operate in different pressure modes.

[0037] Fig. 21 depicts a Force vs. Time graph illustrating a “Hair Trigger” style of Soft Press for FSR-based input.

[0038] Fig. 22 depicts a Force vs. Time graph illustrating a “Hip Fire” style of Soft Press for FSR-based input.

15 [0039] Fig. 23 depicts the controller of Fig. 1 having various sensors disposed within the controller body.

[0040] Fig. 24 is a flow diagram of an example process for re-calibrating a FSR of a handheld controller based on touch data provided by a touch sensor.

[0041] Fig. 25 is a flow diagram of an example process for ignoring spurious input at a FSR of a handheld controller based on touch data provided by a touch sensor for an adjacent control.

20 [0042] Fig. 26 is a flow diagram of an example process for adjusting a FSR input threshold for a FSR based on a hand size detected by an array of proximity sensors in the handle of the handheld controller.

[0043] Fig. 27 is a flow diagram of an example process for activating and deactivating a binding for a control of a handheld controller based on FSR input values.

25 [0044] Fig. 28 is a flow diagram of an example process for using a time-delay to determine whether to ignore FSR input for a first of multiple thresholds.

[0045] Fig. 29 illustrates example components of a handheld controller, such as the controller of Fig. 1.

[0046] Fig. 30 illustrates an example series of gestures that may be performed using the handheld controller of Fig. 1. As illustrated, a user may operate the handheld controller to cause a display device to depict a virtual hand of a user holding and thereafter releasing a virtual object.

30 [0047] Fig. 31 illustrates example components of the handheld controller and/or display devices that may couple to the controller. The controller and/or one or more of the display devices may store or otherwise have access to an object-interaction module and other components to enable the example series of gestures of Fig. 31 and other example gestures.

35 [0048] Figs. 32-35 collectively illustrate an example process that the object-interaction module may implement for causing a display device to present the grasping and releasing of virtual objects.

[0049] Fig. 36 illustrates different calculations that a drop-object module may make, with these calculations being used to determine how to present the release of a virtual object, such as the speed, trajectory, landing location, and/or launch position of the virtual object.

40 [0050] Fig. 37A illustrates an example where the drop-object module calculates a velocity at of the virtual object upon release, with the calculated velocity based at least in part on a peak velocity that occurs within a

predetermined time range prior to determining that the user intends to release the virtual object.

[0051] Fig. 37B illustrates an example where the drop-object module determines a position of the virtual object at a time at which the user intended to release the object, as well as the drop-object module presenting the virtual object being released from that position.

5 [0052] Fig. 37C illustrates an example where the drop-object module calculates a trajectory of the virtual object if the virtual object had been released at the time the user intended to release the virtual object, and presenting the virtual object travelling along this trajectory, albeit beginning at a current position of the virtual object on the display.

[0053] Fig. 37D illustrates an example where the drop-object module calculates a landing location of the virtual object based upon a trajectory of the virtual object if the object had been released at the time the user intended to release the virtual object.

15 [0054] Fig. 38 illustrates an example process of determining, at a first time, that a virtual hand is to release a virtual object and presenting, on a display, the virtual hand releasing the virtual object based at least in part on a peak velocity of the virtual object within a predetermined time range preceding the first time, or a position of the virtual object associated with the peak velocity.

[0055] Fig. 39 illustrates an example process of determining, at a first time, that a virtual object is to be released and presenting, on a display, the virtual object being released based at least in part on a peak velocity of the virtual object within a predetermined time range preceding the first time, or a position of the virtual object associated with the peak velocity.

20 DETAILED DESCRIPTION

[0056] Techniques and devices for holding and releasing virtual objects, presented on a display, based on input received from one or more handheld controllers are described herein. In some instances, a handheld controller includes one or more sensors, such as proximity sensors, force sensors (e.g., force resisting sensors, etc.), accelerometers, and/or other types of sensors configured to receive input from a hand of a user gripping and/or moving the handheld controller. Hardware, software, and/or firmware on the controller and/or on a device coupled to the controller (e.g., a game console, a server, etc.) may receive data from these sensors and generate a representation of a corresponding gesture on a display, such as a monitor, a virtual-reality (VR) headset, and/or the like.

25 [0057] In one example, the sensor data is used to determine when a user is attempting to pick up a virtual object presented on the display. For example, a system (e.g., console, server, etc.) may present one or more virtual objects on a display as part of a game or other type of application. The user may utilize a controller in his or her hand to attempt to hold the virtual object—that is, to cause the system to present a representation of a virtual hand of the user picking up the virtual object. After presenting the virtual hand holding the virtual object, additional sensor data may be used to determine when the user intends to release the virtual object. Upon identifying sensor data indicative of release of the object, the system may cause presentation of the object being released (e.g., dropped, thrown, etc.).

35 [0058] In one example, the handheld controller includes a controller body that may be selectively gripped by the user. The controller body may include an array of proximity sensors configured to detect proximity of fingers of the user based on capacitance values or other sensor readings. In addition, the controller may include one or more force sensors (e.g., force resisting sensors) configured to determine an amount of force on various locations of the

controller body. The controller may also include an accelerometer or other sensor configured to generate velocity data indicative of varying speeds of the handheld controller as the user moves his or her hand.

[0059] The hardware and/or software on the controller or on a device wired or wirelessly coupled to the controller may receive these capacitance values, force values, and/or velocity values to identify different user gestures. For example, if the user is not currently “holding” a virtual object, then upon receiving these varying types of sensor data, the hardware and/or software may attempt to determine when the sensor data indicates the user intends to hold a virtual object. In some instances, the hardware and/or software may determine when the sensor data meets the criteria for one or more different types of hold states, such as a “force-pinch state”, a “force-grip state”, a “capacitive-pinch (cap-pinch) state”, and/or a “cap-grip state”. In some instances, the hardware and/or software may identify a force-pinch state in response to one or more force sensors generating force values that are over a force threshold and one or more buttons being depressed. For example, if a force sensor located on a top side of the handheld controller beneath where a thumb of the user sits returns a force value greater than a particular threshold (e.g., 1% of a maximum force reading of the sensor, 25% of a maximum force reading, etc.) and if a trigger button selectable by a pointer or other finger of the user is pressed, then the described techniques may identify the force-pinch state. That is, in response to these sensor readings, the hardware and/or software may determine that the user is attempting to hold a virtual object via a “pinch”. Meanwhile, the software and/or hardware may identify a force-grip state in response to one or more force sensors on a controller body of the handheld controller measuring one or more force values greater than a force threshold (e.g., 10% of the maximum force reading of the sensor, 25% of the maximum force reading of the sensor, etc.).

[0060] In still other examples, the hardware and/or software may identify a cap-pinch state based on one or more proximity values (e.g., capacitive value) measured by one or more of the proximity sensors, such as the values measured by a proximity sensor on a top portion of the controller where a thumb of the user sits and by a proximity sensor on a controller body of the controller where an index or other finger of the user sits. In some instances, the software and/or hardware may average these two readings and compare the average to a threshold. If the average value is greater than the threshold (e.g., 30% of the maximum sensor reading, 60% of the maximum sensor reading, etc.), then a cap-pinch state may be identified. In other instances, the software and/or hardware may compare a greatest or least reading of these two values and compare that number to the threshold. In yet another example, a cap-grip hold may be identified in based on the proximity values recorded by one or more proximity sensors residing on a controller body of the controller. For example, the proximity values measured by proximity sensors corresponding to a middle, ring, and pinky finger of the user may be averaged and compared to a threshold. If this average is greater than a threshold (e.g., 50% of the maximum sensor reading, 70% of the maximum sensor reading, etc.), then a cap-grip state may be identified. In other instances, the software and/or hardware may compare a greatest or least reading of these two values and compare that number to the threshold. Of course, while a few example hold states are described, other types of hold states may be employed.

[0061] Upon determining that the received sensor data indicates a hold state, the hardware and/or software may store an indication that a particular virtual object is now being held. For example, the hardware and/or software may determine the location of the virtual hand of the user in the application that is being executed, as well as determine the closest virtual object to the location of the virtual hand, and may store an indication that this virtual object is now being held by the user. The hardware and/or software may then proceed to cause presentation, on the display, of the virtual object being held by the virtual hand of the user according to the determined hold type (e.g., pinch, grip, etc.).

While the presented virtual hand may comprise a representation of a human hand, it is to be appreciated that a “virtual hand” may take the form of any other mechanism capable as being displayed to hold an item (e.g., a paw or claw, a tool, etc.). Furthermore, “holding” a virtual object may include a “virtual hand” directly holding an object or otherwise controlling an object, such as causing the object to move by itself in virtual space, but under control of the user via the handheld controller.

[0062] In instances where the hold type is “force-pinch” or “cap-pinch”, the representation may correspond to the virtual hand pinching the virtual object with a thumb and forefinger (for example) of the user. In instances where the hold type is force-grip or cap-grip, the representation may correspond to the virtual hand gripping the virtual object (e.g., in a palm of the virtual hand of the user with the virtual fingers wrapped around the object).

[0063] After the hardware and/or software determines that the virtual object is being held, it may continue to monitor the sensor data to determine when the virtual object is to be released. For example, the hardware and/or software may analyze force data, proximity data, and/or velocity data to determine when the user intends to initiate a release of the item. Upon identifying the initiation of the release, the hardware and/or software may then analyze the sensor data to determine when to execute the release—for example, when to represent the virtual object as being dropped, thrown, or the like. In some instances, velocity data, capacitance data, force data, and/or the like may be analyzed to make this determination. Further, if the hardware and/or software does not identify sensor data indicating the release of the object, then the virtual object may be released a predetermined amount of time after initiation of the release (e.g., measured in seconds, number of frames, etc.). In each instance, the virtual object may be presented on the display as being released of the virtual hand of the user.

[0064] Of course, while the above example describes the object being thrown or dropped, in other instances the techniques described herein may utilize the sensor data to generate other virtual gestures. For example, the force data may be used to generate a representation of a virtual object being smashed in response to the user gripping the controller body with a threshold amount of force.

[0065] In addition to the above, the techniques described herein may determine how to present the release of the virtual object based at least in part on the received sensor data. For example, when the hardware and/or software determines that the virtual object is to be released, the hardware and/or software may calculate one or more of a velocity of the object (e.g., speed and direction), a position at which the virtual object is to be released, a trajectory of the virtual object from the position of release to the landing or other cessation point of the virtual object, a landing location of the virtual object, and/or the like.

[0066] In some instances, the hardware and/or software may determine that a user has intended to release an object at a point in time that is, for example, slightly after the actual time intended by the user. That is, given that the hardware and/or software analyzes sensor data from the handheld controller after this sensor data has been generated, a lag may exist between the intention to release the object expressed by the user holding the handheld controller and the determination of the intention. The techniques may thus attempt to lessen the effect of this lag.

[0067] To do so, upon determining that the sensor data indicates that a virtual object is to be released, the hardware and/or software described herein may perform a “look back” to determine the time (e.g., the frame) at which point the user likely intended to release the object. For example, upon recognizing, at a first time, the initiation of release of the object, the hardware and/or software may analyze sensor data from a predetermined amount of time prior to the first time. This predetermined amount of time may be defined in frames (e.g., a preceding five frames, fifteen frames, fifty frames, etc.), in terms of raw time (e.g., a preceding five milliseconds,

five seconds, etc.), or in any other manner.

[0068] In one specific example, the hardware and/or software may analyze sensor data associated with the prior fifteen frames to determine when the user likely intended to initiate the release. In some instances, the hardware and/or software may analyze the magnitude of the velocity of the handheld controller at each of the preceding fifteen frames to identify a peak velocity during this time range. The frame associated with the peak velocity may be designated at corresponding to the time that the user intended to release the object. Upon identifying this frame, the hardware and/or software may present, on the display, the release of the virtual object based at least in part on information associated with this frame and/or the sensor data associated with this frame.

[0069] In one example, after identifying the frame associated with the peak velocity, the hardware and/or software may calculate an average velocity of this frame, the frame prior to this frame, and the frame after to this frame. The average velocity may comprise both an average speed between the three frames as well as an average direction. The hardware and/or software may then present, on the display, the release of the virtual object based at least in part on this average velocity. For instance, the hardware and/or software may present the virtual object leaving the virtual hand at the average speed and at the average direction of the calculated average velocity.

[0070] In addition, or in the alternative, the hardware and/or software may present the virtual object being released at a position of the virtual object at the time associated with the peak velocity. That is, while the virtual object may be a first position at a time when the hardware and/or software determined that the user intended to initiate the release, the virtual object may have been located at a second, different position at the time that the peak velocity (within the predetermined time range of the first time) occurred. The virtual object may therefore be presented as being released at this second, prior position.

[0071] In still other instances, the hardware and/or software may calculate a landing location of the virtual object had the virtual object been released at the time intended and may display the virtual object landing at this location. That is, the upon identifying the frame associated with the peak velocity, the hardware and/or software may use the peak velocity and the position of the virtual object at this frame to calculate a trajectory of the virtual object had the virtual object been released at that moment. This trajectory may end at a landing location (e.g., a point in the ground, a point on a target, etc.). The hardware and/or software may then calculate a trajectory between a current position of the virtual object at the current time and the landing location and may cause present, on the display, the virtual object traveling at this calculated trajectory. In still other examples, meanwhile, the hardware and/or software may present, on the display, the virtual object travelling along the trajectory that would have occurred if the object were released at the point of peak velocity, albeit with that trajectory starting at the current position of the virtual object at the current time.

[0072] Of course, while a few examples of determining a velocity, trajectory, landing location, or launch position of a virtual object are described above and discussed below with reference to Figs. 36-39, it is to be appreciated that the described hardware and/or software may present a virtual object being released in multiple other manners.

[0073] In addition to the above, force sensing resistors (FSRs) for generating these force values are also described herein. An FSR as described herein may be constructed with a first substrate made of polyimide disposed underneath a second substrate that is resistive and flexible. The first substrate has conductive material (e.g., a plurality of interdigitated metal fingers) disposed on its front surface. One or more spacer layers are also interposed between the first substrate and the second substrate so that a center portion of the second substrate is

suspended over the first substrate. An actuator is disposed on the second substrate to convey an applied force onto a front surface of the second substrate. When this occurs, the center portion of the second substrate flexes inward toward the first substrate, and some of the resistive material on the back surface of the second substrate comes into contact with some of the conductive material on the front surface of the first substrate. As the applied force increases, the surface area of the conductive material that is contacted by the resistive material increases. Likewise, as the applied force decreases, the surface area of the conductive material that is contacted by the resistive material decreases. This change in surface area contact under variable applied force causes the FSR to act as a variable resistor whose value is controlled by the applied force.

[0074] Due at least partly to the polyimide material used for the first substrate, the disclosed FSR exhibits characteristics that make it desirable for use in a controller of a VR system, among other possible end-use applications. For instance, the polyimide substrate allows for selectively soldering the output terminals (or leads) of the FSR directly onto a board (e.g., a PCB) without the use of a bulky header connector, which allows for a FSR with a smaller footprint, as compared to mylar-based FSRs that require a large, bulky header connector. Because polyimide is commonly used as a material of choice for flex circuits, the polyimide substrate of the FSR allows for conveniently connecting the FSR to other flex circuits, which may reduce the cost of manufacturing the disclosed FSR, as compared to the cost of manufacturing conventional FSRs. Polyimide can also withstand high temperatures, such as those of a reflow oven, opening the door to cost-saving manufacturing processes. In addition, polyimide – when used as the first substrate of the disclosed FSR – exhibits desirable characteristics, such as less hysteresis and higher repeatability, as compared to conventional FSRs. Overall, the disclosed FSR, having a first substrate made of polyimide, exhibits a Force vs. Resistance response curve that models a true analog input, making the FSR desirable for use in a controller of a VR system.

[0075] Also disclosed herein is a controller for an electronic system (e.g., a VR system) that includes the disclosed FSR having a first substrate made of polyimide. The controller may be configured to be held by a hand of a user and may include a controller body. The disclosed FSR can be mounted on a planar surface of a structure within the controller body, such as a structure that is mounted within a handle of the controller body, or a structure that is mounted underneath at least one thumb-operated control that is included on a head of the controller body. The FSR, when implemented in the controller for the electronic system, is configured to measure a resistance value that corresponds to an amount of force applied to an associated portion of the controller (e.g., a force applied to an outer surface of the handle, to at least one thumb-operated control, etc.).

[0076] Implementing the FSR in a controller for a VR system allows for expanding the spectrum of natural interaction beyond its current state using conventional controllers. For example, the electronic system and/or the controller can determine, via the FSR, a force with which a user squeezes the handle of the controller, and/or a force with which the user presses a thumb-operated control. Because the disclosed FSR exhibits a desirable response curve, such a controller can translate presses or squeezes of varying force into a varying digitized numerical value that can be used for a video game to control a game mechanic (e.g., to crush a rock, to squeeze a balloon, to toggle through available weapons usable by a game character, etc.). A FSR with desirable response characteristics can replace conventional mechanical switches in order to reduce fatigue of the user and/or to reduce accidental actuation of the controls. For instance, the FSR can act as a switch by detecting when an applied force exceeds a threshold. This threshold adjusted dynamically. For example, the threshold can be adjusted to a lower value in order to reduce hand fatigue during gameplay (e.g., when the user is pressing a

control associated with the FSR to shoot a weapon frequently during gameplay). Conversely, the threshold can be adjusted to a higher value in order to reduce the instances of accidental control operation, which may be useful in a thrilling or exciting game where the user might react to stimuli in the video game.

[0077] Also disclosed herein is a handheld controller that includes logic to implement sensor fusion algorithms based on force data provided by a FSR of the controller in combination with touch data or proximity data provided by a touch sensor or an array of proximity sensors, respectively. An example sensor fusion algorithm can be used to re-calibrate the FSR when an object contacts control associated with the FSR, as detected by the touch sensor. For instance, the logic may determine, based on touch data provided by the touch sensor, that an object has come into contact with a control on the controller body that is configured to be pressed. The logic may also determine, based on force data provided by the FSR at a time at which the object has come into contact with the control, a resistance value measured by the FSR, and may correlate the resistance value with a digitized FSR input value of zero in order to “re-calibrate” the FSR upon detecting a touch at the control.

[0078] Another example sensor fusion algorithm can be used to ignore spurious inputs detected by the FSR when an object is in contact with an adjacent control. For instance, the logic may determine, based on force data provided by the FSR, a resistance value measured by the FSR that corresponds to a digitized FSR input value which meets or exceeds a threshold value that is to be met in order to register a FSR input event for a first control of the handheld controller. The logic may also determine, based on touch data provided by the touch sensor at a time at which the FSR resistance value is measured by the FSR, that the object is in contact with a second control of the handheld controller that is adjacent to the first control, and may refrain from registering the FSR input event while the object is in contact with the second control.

[0079] Another example sensor fusion algorithm can be used to detect a hand size of a hand grasping a handle of the controller, as detected by the array of proximity sensors, and to adjust the threshold force to register a FSR input event at the FSR according to the hand size. This may be useful for making force-based input easier for users with smaller hands (and harder, but not difficult, for users with larger hands). For instance, an array of proximity sensors that are spatially distributed on a handle of the handheld controller can be used to determine a size of a hand that is grasping the handle, and the logic may adjust, based on the size of the hand, a threshold value to an adjusted threshold value that is to be met in order to register a FSR input event for the handle.

[0080] Figs. 1-4 depict a controller 100 for an electronic system according to an example embodiment of the present disclosure. The controller 100 may be utilized by an electronic system such as a VR video gaming system, a robot, weapon, or medical device. The controller 100 may include a controller body 110 having a handle 112, and a hand retainer 120 to retain the controller 100 in the hand of a user (e.g. the user’s left hand). The handle 112 comprises a tubular housing that may optionally be substantially cylindrical. In this context, a substantially cylindrical shape need not have constant diameter, or a perfectly circular cross-section.

[0081] In the embodiment of Figs. 1-4, the controller body 110 may include a head (between the handle 112 and a distal end 111), which may optionally include one or more thumb-operated controls 114, 115, 116. For example, a tilting button, or any other button, knob, wheel, joystick, or trackball may be considered as a thumb-operated control if it may be conveniently manipulated by a user’s thumb during normal operation while the controller 100 is held in the hand of the user.

[0082] The controller 100 preferably includes a tracking member 130 that is fixed to the controller body 110, and optionally includes two noses 132, 134, each protruding from a corresponding one of two opposing distal ends

of the tracking member 130. In the embodiment of Figs. 1-4, the tracking member 130 is preferably but not necessarily a tracking arc having an arcuate shape. The tracking member 130 includes a plurality of tracking transducers disposed therein, preferably with at least one tracking transducer disposed in each protruding nose 132, 134. Additional tracking transducers may be disposed also in the controller body 110, with preferably at least one
5 distal tracking transducer disposed adjacent the distal end 111.

[0083] The foregoing tracking transducers may be tracking sensors that are responsive to electromagnetic radiation (e.g. infrared light) emitted by the electronic system, or they may alternatively be tracking beacons that emit electromagnetic radiation (e.g. infrared light) that is received by the electronic system. For example, the electronic system may be a VR gaming system that widely broadcasts, i.e. paints, pulsed infrared light towards the
10 controller 100, with the plurality of tracking transducers of the tracking member 130 being infrared light sensors that may receive or be shadowed from the broadcast pulsed infrared light. The tracking transducers in each nose 132, 134 (e.g. 3 sensors in each nose) preferably overhang the user's hand on each distal end of the tracking member 130, and so are better exposed (around the user's hand) to receive electromagnetic radiation emitted by the electronic system or to transmit the electromagnetic radiation to the electronic system, at more angles without an unacceptable
15 amount of shadowing.

[0084] In some instances, the tracking member 130 and the controller body 110 are made of a substantially rigid material such as hard plastic, and are firmly fixed together so that they do not appreciably translate or rotate relative to each other. In this way, the tracking of the translation and rotation of the constellation of tracking transducers in space, is preferably not complicated by motion of the tracking transducers relative to each other.
20 For example, as shown in Figs. 1-4, the tracking member 130 may be fixed to the controller body 110 by being joined to the controller body 110 at two locations. The hand retainer 120 may be attached to the controller 100 (either the controller body 110 or the tracking member 130) adjacent those two locations, to bias the user's palm against the outside surface of the handle 112 between the two locations.

[0085] In certain embodiments, the tracking member 130 and the controller body 110 may comprise an integral monolithic component having material continuity, rather than being assembled together. For example, the tracking member 130 and the controller body 110 may be molded together by a single injection-molding process step, resulting in one integral hard plastic component that comprises both the tracking member 130 and the controller body 110. Alternatively, the tracking member 130 and the controller body 110 may be initially fabricated separately, and then later assembled together. Either way, the tracking member 130 may be considered as fixed to the controller
30 body 110.

[0086] The hand retainer 120 is shown in the open position in Fig. 1. The hand retainer 120 may optionally be biased in the open position by a curved resilient member 122, to facilitate the insertion of the user's left hand between the hand retainer 120 and the controller body 110 when the user is grasping for the controller with vision blocked by VR goggles. For example, the curved resilient member 122 may optionally be a flexible metal strip that elastically bends, or may comprise an alternative plastic material such as nylon that may bend substantially
35 elastically. The curved resilient member 122 may optionally be partially or completely internal to or covered by a cushion or fabric material 124 (e.g. a neoprene sheath), for the user's comfort. Alternatively, the cushion or fabric material 124 may be disposed on (e.g. adhered to) only the side of the curved resilient member 122 that faces the user's hand.

[0087] The hand retainer 120 optionally may be adjustable in length, for example by including a draw cord 126
40

that is cinched by a spring-biased chock 128. The draw cord 126 may optionally have an excess length that may be used as a lanyard. The sheath 124 optionally may be attached to the draw cord. In certain embodiments, the curved resilient member 122 may be preloaded by the tension of the cinched draw cord 126. In such embodiments, the tension that the curved resilient member 122 imparts to the hand retainer 120 (to bias it in the open position) causes the hand retainer to automatically open when the draw cord 126 is un-cinched. This disclosure also contemplates alternative conventional ways to adjust the length of a hand retainer 120, such as a cleat, an elastic band (that temporarily stretches when the hand is inserted, so that it applies elastic tension to press against the back of the hand), a hook & loop strap attachment that allows length adjustment, etc.

[0088] The hand retainer 120 may be disposed between the handle 112 and the tracking member 130, and be configured to contact the back of the user's hand. Fig. 2 shows the controller 100 during operation with the user's left hand inserted therein but not grasping the controller body 110. In Fig. 2, the hand retainer 120 is closed and tightened over the hand, to physically bias the user's palm against the outside surface of the handle 112. In that way, the hand retainer 120, when closed, may retain the controller 100 to the hand even when the hand is not grasping the controller body 110. Figs. 3 and 4 depict the controller 100 during operation when the hand retainer 120 is closed, and the hand is grasping the controller body 110 and the thumb is operating one or more of the thumb-operated controls (e.g. track pad 116).

[0089] The handle 112 of the controller body 110 preferably includes an array of proximity sensors that are spatially distributed partially or completely around its outer surface. The proximity sensors of the array are not necessarily of equal size and do not necessarily have equal spacing between them, although the array may comprise a grid. The array of proximity sensors is preferably responsive to the proximity of the user's fingers to the outside surface of the handle 112. For example, the array of proximity sensors may be a plurality of capacitive sensors embedded under the outer surface of the handle 112, with that outer surface comprising an electrically insulative material. The capacitance between such an array of capacitive sensors and a portion of the user's hand is inversely related to the distance there between. The capacitance may be sensed by connecting an RC oscillator circuit to an element of the capacitance sensor array, and noting that the time constant of the circuit (and therefore the period and frequency of oscillation) will vary with the capacitance. In this way, the circuit may detect a release of a user's fingers from the outer surface of the handle 112.

[0090] When the hand retainer 120 (e.g. a hand-retention strap) is closed tightly, it may serve not only to prevent the controller 100 from falling out of hand, but also to keep fingers from excessively translating relative to the proximity sensor array of the handle 112, to more reliably sense finger motion. The electronic system may include an algorithm embodying anatomically-possible motions of fingers, to better use the sensing from the proximity sensor array to render the opening of a controlled character's hand, finger pointing, or other motions of fingers relative to controller or relative to each other. In this way, the user's movement of the controller 100 and/or fingers may help control a VR gaming system, defense system, medical system, industrial robot or machine, or another device. In VR system applications (e.g. for gaming, training, etc.), the system may render a throwing motion based on the movement of the tracking transducers, and may render the release of a thrown object based on the sensed release of the user's fingers from the outer surface of the handle of the controller.

[0091] Hence, the function of the hand retainer 120 (to allow the user to "let go" of the controller 100 without the controller 100 actually separating from the hand or being thrown or dropped to the floor) may enable additional functionality of the controlled electronic system. For example, if the release and restoration of the user's grasp of

the handle 112 of the controller body 110 is sensed, then such release or grasping may be incorporated into the game to display (e.g. in VR) throwing or grasping objects. The hand retainer 120 may allow such a function to be accomplished repeatedly and safely. For example, the location of the hand retainer 120 in the embodiment of Figs. 1-4 may help the tracking member 130 to protect back of user's hand from impacts in real world, for example when the user moves in response to a prompt sensed in the VR environment (e.g. while practically blinded by VR goggles).

[0092] In certain embodiments, the controller 100 may include a rechargeable battery disposed within the controller body 110, and the hand retainer 120 (e.g. hand retention strap) may include an electrically-conductive charging wire that is electrically coupled to the rechargeable battery. The controller 100 preferably also includes a radio frequency (RF) transmitter for communication with the rest of the electronic system. Such RF transmitter may be powered by the rechargeable battery and may be responsive to the thumb-operated controls 114, 115, 116, the proximity sensors in the handle 112 of the controller body 110, and/or tracking sensors in the tracking member 130.

[0093] As shown in Fig. 5, in certain embodiments the controller 100 may be the left controller in a pair of controllers that includes a similar right controller 200. In certain embodiments, the controllers 100 and 200 may (together) track the motion and grip of both of a user's hands, simultaneously, for example to enhance a VR experience.

[0094] Fig. 6A depicts a front view of right-hand controller 600 according to another example embodiment of the present disclosure. Fig. 6B depicts a back view of the right-hand controller 600. The controller 600 has a controller body comprising a head 610 and a handle 612. In the embodiment of Figs. 6A-6B, the head 610 includes at least one thumb-operated control A, B, 608, and may also include a control configured to be operated by the index finger (e.g. trigger 609). The handle 612 comprises a tubular housing that is partially wrapped by an outer shell 640.

[0095] In the embodiment of Figs. 6A-6B, a tracking member 630 is fixed to the controller body at the head 610 and at an end of the handle 612. A hand retainer 620 is configured to physically bias the user's palm against the outer shell 640 between the head 610 and the end of the handle 612. The hand retainer 620 is preferably disposed between the handle 612 and the tracking member 630, and may comprise a hand retention strap that is adjustable in length and configured to contact the back of the user's hand. In the embodiment of Figs. 6A-6B, the hand retainer 620 optionally includes a draw cord 628, and optionally can be adjusted in length by a cord lock 626 (adjacent a distal end of the handle 612) that selectively prevents sliding motion by the draw cord 628 at the location of the cord lock 626.

[0096] In the embodiment of Figs. 6A-6B, tracking transducers 632, 633 are disposed on the tracking member 630, with tracking transducers 633 being disposed on protruding noses at opposing distal ends of the tracking member 630. Additional tracking transducers 634 are optionally disposed on a distal region of the head 610. The tracking transducers 632, 633, and 634 may be tracking sensors that are responsive to electromagnetic radiation (e.g. infrared light) emitted by the electronic system (e.g. virtual reality gaming system), or may be tracking beacons that emit electromagnetic radiation (e.g. infrared light) that is received by the electronic system. For example, the electronic system may be a VR gaming system that widely broadcasts, i.e. paints, pulsed infrared light towards the controller 600, with the tracking transducers 632, 633, and 634 being infrared light sensors that may receive the broadcast pulsed infrared light. The response of such tracking sensors may be communicated back to the electronic system, and the system may interpret such response to effectively track the location and orientation of

the controller 600.

[0097] One or more of the tracking transducers 632, 633, 634 optionally may be structured as shown in the embodiment of Fig. 7A, or alternatively shown in the embodiment of Fig. 7B, or alternatively in a conventional way that is not shown. The lower portion of Fig. 7A depicts an exploded perspective view of an infrared light sensor 750 that is electrically connected to a flex circuit 751, shown beneath a rectangular portion of an overlying windowed housing wall 755 that comprises an infrared-opaque plastic. The windowed housing wall 755 includes a window 756. The window 756 preferably comprises an infrared-transmissive polycarbonate plastic, and may include an underside recession to accommodate the thickness of the infrared light sensor 750.

[0098] According to the embodiment of Fig. 7A, the windowed housing wall (e.g. the outer structure of the tracking member 630, or the head 610 of Fig. 6A) may be fabricated from a so-called “double shot” injection molding process, so that the majority of the housing wall is fabricated from infrared-opaque plastic, but with infrared-transmissive plastic being disposed in the window 756 above the infrared light sensor 750.

[0099] The upper portion of Fig. 7A depicts a cross-sectional view of the infrared light sensor 750, flex circuit 751, and the windowed housing wall 755 as assembled. Infrared light, shown in Fig. 7A as three downward arrows incident upon the window 756 from above, passes through the window 756 to be received by the underlying infrared light sensor 750. Since the housing wall 755 comprises infrared-opaque plastic, the infrared light that strikes it will not pass through, and a portion may be reflected back into the window to be received by the infrared light sensor 750. In this way, the window 756 permits infrared light to affect the infrared light sensor 750, despite the majority of the housing wall 755 comprising infrared-opaque plastic, so that the infrared light sensor 750 receives infrared light only from a preferred angular range.

[0100] Alternatively, one or more of the tracking transducers 632, 633, 634 optionally may be structured as shown in the embodiment of Fig. 7B. The lower portion of Fig. 7B depicts an exploded perspective view of the infrared light sensor 750 as electrically connected to the flex circuit 751, shown beneath a rectangular portion of an overlying housing wall 758 that comprises an IR-transmissive plastic. The housing wall 758 is coated with an infrared-opaque film 757 that is patterned to include a window 759 (where the infrared-opaque film 757 is absent).

[0101] The upper portion of Fig. 7B depicts a cross-sectional view of the infrared light sensor 750, flex circuit 751, the housing wall 758, and the IR-opaque film 757, as assembled. Infrared light, shown in Fig. 7B as three downward arrows incident upon the housing wall 758 from above, passes through the window 759 in the infrared-opaque film 757 to pass through the housing wall 758 there to be received by the underlying infrared light sensor 750. Since the housing wall 758 comprises infrared-transmissive plastic, the infrared light that strikes it may pass into it and be lost, and perhaps unintentionally and undesirably even reach a nearby sensor via internal reflections. In this way, the window 759 in the infrared-opaque film 757 permits infrared light to primarily affect the infrared light sensor 750.

[0102] Fig. 8 shows a side view of the right-hand controller 600, with the outer shell 640, which partially wraps the tubular housing of the handle 612 being exploded away to reveal instrumentation on its inner surface. In the embodiment of Fig. 8, the instrumentation may comprise an array of proximity sensors 800 that are spatially distributed on the inner surface of the outer shell 640, the array of proximity sensors 800 being responsive to a proximity of the user's fingers to the outer shell 640. The proximity sensors 800 of the array are not necessarily of equal size, nor are they necessarily spaced regularly or equally from each other. In certain embodiments, the array of proximity sensors 800 preferably may be a plurality of capacitive sensors that may be connected to a flex circuit that

is bonded to the inner surface of the outer shell 640. In the embodiment of Fig. 8, the outer shell 640 includes a first electrical connector portion 805, which may be connected to a mating second electrical connector portion of the handle 612 (as shown in more detail in Figs. 9A-9B).

[0103] Figs. 9A-B depicts cross sections of the right-hand controller 600 of Fig. 6A, showing that the controller's handle optionally may comprise a tubular housing 612a, 612b, that is split longitudinally by a seam 613 where the tubular housing portions 612a and 612b adjoin. In Fig. 9A, the outer shell 640 is shown exploded away from the rest of the handle. Fig. 9B depicts the cross section of Fig. 9A, except with the outer shell 640 installed in its normal operational position. In the embodiment of Figs. 9A-9B, the first electrical connector portion 805 of the outer shell 640 is shown to be mating and connectable to the second electrical connector portion 905 of the controller handle.

[0104] In the embodiment of Figs. 9A-9B, the outer shell 640 partially wraps the tubular housing 612a, 612b in such a way that it preferably overlaps the longitudinal seam 613, so that the longitudinal seam 613 may be positioned to optimize the process of manufacture rather than to accommodate the desired circumferential location of the proximity sensor array 800. In certain embodiments, the outer shell 640 overlaps a circumferential portion C of the tubular housing 612a, 612b of the handle, and the circumferential portion C angularly spans at least 100 degrees but not more than 170 degrees of the full circumference of the tubular housing 612a, 612b of the handle. Such a circumferential overlap may, in certain embodiments, enable the proximity sensor array 800 to sense the proximity of a desired portion of the user's fingers or palm, for example the region of the hand that best indicates grasping.

[0105] The tubular housing 612a, 612b of the handle need not have a circular cross-section, and that the word "circumference" is used herein whether or not the tubular housing 612a, 612b of the handle has a circular cross-section. Herein, the term "circumference" implies the complete perimeter about the tubular housing 612a, 612b of the handle, which may be circular if the tubular housing 612a, 612b is a right circular hollow cylinder, but which may be a closed shape other than a circle if the tubular housing is shaped as a non-circular cylinder or hollow prism.

[0106] In the embodiment of Figs. 9A-9B, a printed circuit board (PCB) 920 may be mounted within the tubular housing 612a, 612b of the handle, with the second electrical connector portion 905 being electrically coupled to the PCB 920. The PCB 920 optionally includes a force sensing resistor (FSR) 922, and the controller may further comprise a plunger 924 that conveys a compressive force applied via the outer shell 640 towards the outside of the tubular housing 612a, 612b of the handle inward to the FSR 922. In certain embodiments, the FSR 922, in conjunction with the proximity sensor array 800, may facilitate sensing of both the onset of grasping by the user, and the relative strength of such grasping by the user, which may be facilitate certain gameplay features.

[0107] In certain embodiments, the outer shell 640 has a shell thickness (measured radially in Figs. 9A-9B) that is less than one-third of a housing wall thickness of the tubular housing portions 612a or 612b of the handle. In those embodiments, such a thickness inequality may improve the sensitivity of the proximity sensor array 800 relative to an alternative embodiment where the proximity sensor array 800 is disposed on or in the tubular housing 612a, 612b of the handle.

[0108] Fig. 10A depicts a front view of right-hand controller 200 according to another example embodiment of the present disclosure, with a partially-closed hand retainer 220 (e.g. a hand retention strap). Fig. 10B depicts a front view the controller 200, except with the hand retainer 220 fully open. In the embodiment of Figs. 10A-10B, the controller 200 includes a controller body having a head 210 and a handle 212. The head 210 adjoins the

handle 212 at a neck region 211 of the controller 200. The handle 212 preferably includes an array of proximity sensors that are spatially distributed just under its outside surface, and that are preferably responsive to a proximity of the user's fingers to the outer surface of the handle 212.

[0109] In the embodiment of Figs. 10A-10B, the head 210 includes thumb-operated controls A, B, and 208.

5 The controller 200 also includes a tracking member 230 that is preferably fixed to the controller body at the head 210 and at a distal end of the handle 212. The tracking member 230 preferably includes a plurality of tracking transducers that may be sensors that are responsive to electromagnetic radiation emitted by the electronic system (e.g. pulsed infrared light emitted by a virtual reality gaming system), or tracking beacons that emit electromagnetic radiation to be received by the electronic system. In the embodiment of Figs. 10A-10B, the
10 tracking member 230 is preferably but not necessarily a tracking arc having an arcuate shape. The hand retainer 220 is preferably disposed between the handle 212 and the tracking arc 230.

[0110] In the embodiment of Figs. 10A-10B, the controller 200 includes a draw cord 228, and a cord lock 226 adjacent a distal end of the handle 212. The cord lock 226 may selectively prevent sliding motion by the draw cord 228 at the cord lock 226. In the embodiment of Fig. 10A, as the draw cord 228 is pulled progressively
15 further past the cord lock 226, the hand retainer 220 is drawn tighter into a closed position (as shown by the motion arrow depicted in Fig. 10A). The closed position physically biases the user's palm against an outer surface of the handle 212.

[0111] In the embodiment of Figs. 10A-10B, the hand retainer 220 preferably includes a resilient member (e.g. an internal or external elastically deformable strip such as a metal strip) that biases the hand retainer 220
20 towards the open position shown in Fig. 10B. In the embodiment of Fig. 10B, when the user selectively causes the cord lock 226 to release and permit relative sliding of the draw cord 228, the preloaded bias towards straightening of the elastically deformed resilient member causes the hand retainer 220 to naturally open (as shown by the motion arrow depicted in Fig. 10B). The open position may facilitate inserting or withdrawing the user's hand from the controller 200, especially when the user's vision may be obstructed by the wearing of virtual
25 reality goggles.

[0112] Fig. 11A depicts a front view of the head 210 and handle 212 components of the controller 200, including a hand retainer anchor 302 that can be adjusted to move peripherally about the head 210. Fig. 11B depicts the same head 210 and handle 212 components, except with a faceplate removed from the head 210 to expose a lockable collar portion 311 that may facilitate selective adjustment of the hand retainer anchor 302
30 peripherally about the head 210.

[0113] In the embodiment of Fig. 11B, the lockable collar portion 311 may translate along an arcuate path defined by an internal arcuate guide 315. The lockable collar portion 311 can be selectively locked by the user to prevent further movement of the anchor 302 about the periphery of the head 210. Now referring to Figs. 4 and 10A-11B, the resilient member of the hand retainer 220 is attached to the hand retainer anchor 302 of the head 210,
35 which permits the hand retainer 220 to be adjusted towards or away from the user's purlicue (between the user's thumb and fingers). In certain embodiments, the resilient member of the hand retainer 220 is preferably attached to the hand retainer anchor 302 of the head 210 by a pivoting or rotatable attachment, so that the hand retainer 220 can pivot relative to the hand retainer anchor 302 at the location of the attachment. Such degree of freedom is additional to the adjustability of the position of the hand retainer anchor 302 about the periphery of the head
40 210.

[0114] Figs. 12A, 12B, and 12C depict an alternative embodiment of a partially assembled controller 400 having a controller body that includes a head 410 and a handle 412 joined to the head in a neck region 411. In the alternative embodiment of Figs. 12A-12C, the controller body includes a channel 414 that is disposed adjacent the neck region 411. A hand retainer, which is not shown in Fig. 12A so that the channel 414 will not be partially obscured, includes a resilient member 420 that terminates in a projection 425 that extends into the channel 414.

[0115] In the embodiment of Figs. 12B and 12C, the projection 425 includes a catch 427 that prevents longitudinal movement of the projection within the channel 414 when the hand retainer is in the closed position. For example, in the embodiment of Fig. 12C, the catch 427 is a cam that increases friction with an interior surface of the channel 414, when a relative angle of the hand retainer projection 425 corresponds to the closed position of the hand retainer – i.e., when the closed position of the hand retainer results in tension upon the resilient member 420 (e.g. in a downward direction as shown in the cross-section of Fig. 12C).

[0116] By contrast, when the hand retainer projection 425 is rotated to a relative angle that corresponds to an open position of the hand retainer (e.g. in an upward direction as shown in the cross-section of Fig. 12C), the friction between the catch 427 and the channel 414 is reduced, and the hand retainer projection 425 may be translated within the channel 414 (as indicated by the motion arrows shown in Fig. 12B). The channel 414 is preferably oriented so that translation of the hand retainer projection along the channel 414 preferably adjusts the relative position of the hand retainer projection 425 towards or away from the purlicue of the user's hand, for example so that the controller 400 can accommodate different hand sizes or finger lengths. In an alternative embodiment, the hand retainer projection 425 may be pivotably attached to the remainder of the hand retainer by a conventional pivot joint. Such rotational degree of freedom is additional to the adjustable translation of the hand retainer projection 425 along the channel 414.

[0117] Figs. 13A-C depict different views of a force sensing resistor (FSR) 1300 according to an example embodiment of the present disclosure. As shown in the cross section of the FSR 1300 in Fig. 13C, the FSR 1300 may include a first substrate 1302 made of polyimide. The FSR 1300 may further include a second substrate 1304 disposed on (or over) the first substrate 1302. The first substrate 1302 and the second substrate 1304 may be considered to be the two primary substrates (or layers) of the FSR 1300, which can be considered a 2-layer FSR 1300, although it is to be appreciated that the FSR 1300 includes additional layers, as will be described in more detail herein. In this context, the first substrate 1302 may be considered a “bottom” or “base” substrate with respect to the two primary substrates of the FSR 1300, although it is to be appreciated that there may be layers of material behind (or below) the first substrate 1302 (i.e., in the negative Z direction, as depicted in Fig. 13C).

[0118] The first substrate 1302 has a conductive material disposed on a front surface (i.e., the surface facing in the positive Z direction) of the first substrate 1302. As will be described in more detail with reference to Fig. 14, this conductive material may include a plurality of interdigitated metal fingers. Meanwhile, the second substrate 1304 (sometimes referred to as a resistive “membrane”) has resistive material disposed on a back surface (i.e., the surface facing the negative Z direction) of the second substrate 1304. This resistive material may be a semiconductive material, such as an ink composition (e.g., silver ink, carbon ink, mixtures thereof, etc.), that exhibits some level of electrical resistance (e.g., a relatively high sheet resistance within a range of 300 kiloOhm (kOhm) per square (kOhm/sq) to 400 kOhm/sq). Preferably, the sheet resistance of the second substrate 1304 is 350 kOhm/sq, although it is to be appreciated that other sheet resistance values, including those outside of the sheet resistance ranges specified herein, may be used, such as when the FSR 1300 is used in other applications,

such as non-controller based applications. As such, the sheet resistance range(s) specified herein is to be understood as non-limiting. In some embodiments, the second substrate 1304 may be made of mylar, with the resistive material disposed on a back surface of the second substrate 1304. In some embodiments, the second substrate 1304 is made of polyimide having a resistive material (e.g., a conductive ink composition) on the back surface. An example benefit of using polyimide for the second substrate 1304 is to create an FSR 1300 that can be mass manufactured using a reflow oven, whereas mylar could not withstand such high temperatures.

[0119] The FSR 1300 may include one or more spacer layers interposed between the first substrate 1302 and the second substrate 1304 so that a center portion of the second substrate 1304 is suspended over the first substrate 1302 and spaced a distance therefrom. Fig. 13C shows two spacer layers including, without limitation, a coverlay 1306 disposed on the first substrate 1302 at a periphery of the first substrate 1302, and a layer of adhesive 1308 disposed on the coverlay 1306. The coverlay 1306 may be made of polyimide, and may thus be the same material as the first substrate 1302. A thickness (as measured in the Z direction) of the coverlay 1306 may be within a range of 10 microns to 15 microns. A thickness (as measured in the Z direction) of the layer of adhesive 1308 may be within a range of 50 microns to 130 microns. The total distance at which the second substrate 1304 is spaced from the first substrate 1302 may, therefore, be the sum of the thicknesses of the one or more spacer layers (e.g., the thickness of the coverlay 1306 plus the thickness of the layer of adhesive 1308). These layers may be provided at thicknesses that are outside of the thickness ranges specified herein, such as when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these thickness ranges are to be understood as non-limiting.

[0120] An actuator 1310 (such as a disk-shaped, compliant plunger) may be disposed on the second substrate 1304, and is configured to convey a force, F , onto a front surface of the second substrate 1304. The actuator 1310 can be made of Poron, which is a compliant material that deforms to a degree upon application of a force upon the actuator 1310. The actuator 1310 may be concentric with a center of an active area of the FSR 1300 in order to center the applied force, F . The actuator 1310 also spans a portion of the active area of the FSR 1300 in order to evenly distribute the applied force, F , across that portion of the active area of the FSR 1300.

[0121] A thickness (as measured in the Z direction) of the second substrate 1304 may be within a range of 50 microns to 130 microns. At this example thickness, the second substrate 1304 is flexible. For example, the second substrate 1304 can be made of mylar, which is flexible at a thickness within the above-specified range. Functional operation of the FSR 1300 relies on the flexibility of the second substrate 1304 in order for the resistive material on the back surface of the second substrate 1304 to come into contact with the conductive material on the front surface of the first substrate 1302 under a compressive force, F , applied to the actuator 1310. A thickness (as measured in the Z direction) of the first substrate 1302 may be within a range of 20 microns to 30 microns. Polyimide, at this thickness, is also flexible. Thus, the first substrate 1302 is also flexible. Meanwhile, a thickness (as measured in the Z direction) of the actuator 1310 may be within a range of 780 microns to 810 microns. These layers may be provided at thicknesses that are outside of the thickness ranges specified herein, such as when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these thickness ranges are to be understood as non-limiting.

[0122] The FSR 1300 may exhibit varying resistance in response to a variable force, F , applied to the actuator 1310. For example, as the force, F , on the actuator 1310 is increased, the resistance is decreased. In this manner, the FSR 1300 may be treated as a variable resistor whose value is controlled by the applied force, F . The

FSR 1300 can be a “ShuntMode” FSR 1300 or a “ThruMode” FSR 1300, but is preferably a ShuntMode FSR 1300. With a ShuntMode FSR 1300, the conductive material disposed on the front surface of the first substrate 1302 may be in the form of a plurality of interdigitated metal fingers. When the force, F , is applied to the front (or top) of the actuator 1310, the resistive material on the back surface of the second substrate 1304 comes into contact with some of the interdigitated metal fingers, which shunts the metal fingers, thereby varying the resistance across the output terminals of the FSR 1300. In a ThruMode implementation, the conductive material on the first substrate 1302 may be a solid area of conductive material with a semiconductive (or resistive) material disposed on the conductive material, and the second substrate 1304 may have a similar construction (e.g., a solid area of conductive material having a semiconductive (or resistive) material disposed thereon). The solid area of conductive material on each substrate (1302 and 1304) is coupled to an individual output terminal, and excitation current can pass through one layer to the other when the two substrates (1302 and 1304) come into contact under an applied force, F .

[0123] In at least the preferred ShuntMode implementation, the Force vs. Resistance response curve – where the FSR 1300 resistance is plotted as a function of applied force, F – exhibits desirable characteristics for use in a controller 100/600 of a VR system. For instance, the response curve of the FSR 1300 may exhibit less hysteresis and higher repeatability (from one FSR 1300 to another FSR 1300), as compared to conventional FSRs, such as those that use mylar as the material for the bottom substrate. Loading hysteresis describes the effect of previously applied forces on the current FSR 1300 resistance. The response curve is also monotonic, and it models a true analog input that can be leveraged for a number of game mechanics in a VR gaming system, such as to crush a virtual rock, squeeze a virtual balloon, etc. It is to be appreciated that although examples herein describe an applied force, F , the FSR 1300 is, in actuality, sensitive to applied pressure (force $\times$ area) because equal amounts of force applied at a small point verses a larger area on front surface of the second substrate 1304 will result in a different resistance response of the FSR 1300. Thus, the actuator 1310 plays a role in maintaining repeatability across FSRs 1300 in terms of the response curves under applied force, F .

[0124] Fig. 14 depicts various front views of the FSR 1300 at progressive stages in an example process of constructing the FSR 1300. At Stage 1 of Fig. 14, a plurality of interdigitated metal fingers 1400 may be formed on a front surface of the first substrate 1302 of polyimide. The metal fingers 1400 are electrically conductive. An example conductive metal used for the metal fingers 1400 is copper, such as 1/3 oz. HA copper. This copper may also be gold plated. The plurality of interdigitated metal fingers 1400 may be formed using a subtractive manufacturing process. For example, prior to Stage 1, the first substrate 1302 of polyimide may be formed with a copper clad layer disposed on its front surface, and the copper clad layer may be etched (e.g., by removing strips of copper material) to create the pattern of interdigitated metal fingers 1400 shown in Stage 1 of Fig. 14. The size and spacing of the etched pattern may be chosen to create a distance (as measured in the Y direction) between pairs of adjacent metal fingers 1400 that is 0.2 millimeters (mm), and a width (as measured in the Y direction) of each metal finger of the plurality of interdigitated metal fingers 1400 that is 0.2 mm. This finger width and spacing between fingers may provide an optimal balance between maximum sensitivity of the FSR 1300 and a minimized manufacturing etch tolerance. Although a uniform pattern of metal fingers 1400 is shown in Fig. 14, it is to be appreciated that other non-uniform patterns (e.g., denser fingers toward a center, and less dense fingers toward the outside) may be employed. Fig. 14 shows two sets of interdigitated metal fingers 1400, each leading to an output terminal 1402 (or lead) of a 2-terminal FSR 1300 having a first output terminal 1402(1) and a second output terminal 1402(2).

[0125] As mentioned, the copper that constitutes the metal fingers 1400 may be gold plated. Thus, after etching the pattern of interdigitated metal fingers 1400, a layer of gold plating may be deposited onto the copper fingers to create gold-plated fingers. Thus, the plurality of interdigitated metal fingers 1400 shown in Stage 1 of Fig. 14 can represent gold-plated fingers. The gold-plating may be electroless nickel immersion gold (ENIG).

5 Notably, there may be no additional copper plating over the base layer copper prior to gold plating. Additional copper plating is commonly applied atop the base layer copper when adding vias to multi-layer flex substrates. However, adding additional copper plating over the base layer copper prior to gold-plating may actually cause an undesirable increase of detected resistance, as compared to the disclosed FSR 1300 that does not include any additional copper plating over the base layer copper prior to gold plating. Thus, the omission of any additional

10 copper plating on the metal fingers 1400 prior to the gold plating achieves optimal sensitivity in the FSR 1300. Thus, the copper clad layer that constitutes the metal fingers 1400 remains exposed at the time that the metal fingers 1400 are plated with gold material. In this manner, the gold material is in direct contact with the base copper material of the metal fingers 1400, without any additional copper plating interposed between the base layer copper and the gold plating.

15 [0126] At Stage 2 of Fig. 14, a coverlay 1306 can be deposited atop the first substrate 1302 at a periphery of the first substrate 1302. For example, the coverlay 1306 can be annular in shape to cover a peripheral portion of the metal fingers 1400, and a remaining portion of the metal fingers 1400 is left uncovered by the coverlay 1306 after deposition. The coverlay 1306 may be made of polyimide.

[0127] At Stage 3 of Fig. 14, a layer of adhesive 1308 may be deposited atop the coverlay 1306 such that the

20 remaining portion of the metal fingers 1400 (the portion of the metal fingers 1400 left uncovered by the coverlay 1306) is also left uncovered by the layer of adhesive 1308. For example, the layer of adhesive 1308 can be C-shaped so that the layer of adhesive 1308 covers a substantial portion of the coverlay 1306, and so that the layer of adhesive 1308 does not cover an active area of the FSR 1300. The “active area” of the FSR 1300 is shown in Stage 3 of Fig. 14 as having a diameter, B. Furthermore, the layer of adhesive 1308, being C-shaped, can leave a

25 section of the coverlay 1306 uncovered by the layer of adhesive 1308. This uncovered section of the coverlay 1306 is shown in Stage 3 of Fig. 14 as having a width, w . After the second substrate 1304 is placed over the top of the first substrate 1302, this uncovered section of coverlay 1306 creates an air gap that allows air to ingress and/or egress from a space between the first substrate 1302 and the second substrate 1304, which may prevent sensor-to-sensor response variations due to changes in atmospheric pressure. The width, w , of the air gap (i.e.,

30 the uncovered section of the coverlay 1306) can be 1 mm, which is small enough to preserve symmetry of the contacted surface area under an applied force, and large enough to allow air to ingress/egress through the air gap. In some embodiments, the layer of adhesive 1308 may be 467 adhesive from 3M® Company of Maplewood, Minnesota (i.e., 3M 467 adhesive). The coverlay 1306 and the layer of adhesive 1308 represent examples of spacer layers that can be provided atop the first substrate 1302 in order to space the second substrate 1304 a

35 distance from the first substrate 1304 in a suspended fashion. As mentioned, a thickness (as measured in the Z direction) of the coverlay 1306 may be within a range of 10 microns to 15 microns, and a thickness (as measured in the Z direction) of the layer of adhesive 1308 may be within a range of 50 microns to 130 microns. Preferably, the thickness of the layer of adhesive 1308 is made as thin as possible (e.g., at the lower end of the specified thickness range) to allow for an initial response (e.g., the FSR 1300 starts detecting an input) under a very light

40 applied force, F. However, these layers may be provided at thicknesses that are outside of the thickness ranges

specified herein, such as when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these thickness ranges are to be understood as non-limiting.

[0128] At Stage 4, a second substrate 1304 can be provided atop the first substrate 1302. In Stage 4, a center portion of the second substrate 1304 is suspended over the first substrate 1302 by virtue of the one or more spacer layers (e.g., the coverlay 1306 and the layer of adhesive 1308) interposed between the first substrate 1302 and the second substrate 1304 (See Fig. 13C). Although it is not shown in Fig. 14, the actuator 1310 can be attached to the front surface of the second substrate 1304 in order to complete the construction of the FSR 1300, as shown in Figs. 13A-C. The size (as measured in the X-Y plane) of the actuator may span 80% of the active area of the FSR 1300 (i.e., 80% of the diameter, B, shown in Stage 3 of Fig. 14). For instance, a disk shaped actuator 1310 may have a diameter that is equal to $0.8*B$. In some embodiments, the overall diameter of the FSR 1300 may be 14.5 mm. At this dimension, the active area may have a diameter, B, of 10.5 mm, meaning that the coverlay 1306 and the layer of adhesive 1308 may be deposited as a 2 mm ring between the first substrate 1302 and the second substrate 1304. In this embodiment, the actuator 1310 may have a diameter of 8.4 mm (i.e., $0.8*10.5$ mm).

[0129] The FSR 1300 may be open-circuit under no external force (or load). In some embodiments, in order to account for any contact of the first substrate 1302 and the second substrate 1304 under zero or negligible applied force, a threshold circuit can be used to set a threshold resistance value at which the first substrate 1302 and the second substrate 1304 are considered to be “in contact,” meaning that the FSR 1300 can be open-circuit until the threshold resistance value is met, even if the two primary substrates (i.e., 1302 and 1304) are actually in contact.

[0130] Fig. 15 depicts example layers of a FSR 1300, according to another embodiment of the present disclosure. Fig. 15 is not to scale. Rather, Fig. 15 is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR 1300. As described above with reference to the previous figures, the FSR 1300, as shown in Fig. 15, includes a first substrate 1302 made of polyimide, metal fingers 1400 (i.e., a conductive material) disposed on the front surface of the first substrate 1302, and a second substrate 1304 disposed on the first substrate 1302, with one or more spacer layers interposed between the first substrate 1302 and the second substrate 1304; in this case, there are multiple spacer layers disposed between the two primary substrates, including the aforementioned coverlay 1306 and the layer of adhesive 1308. An actuator 1310 is also disposed on the second substrate 1304.

[0131] In the embodiment of Fig. 15, the actuator 1310 may be made of Poron, and may have a thickness (as measured in the Z direction) of 794 microns. A layer of actuator adhesive 1500 may be used to attach the actuator 1310 to the second substrate 1304. This actuator adhesive 1500 can be 70 microns in thickness (as measured in the Z direction). A suitable adhesive for the actuator adhesive 1500 is FT 8397 adhesive from Avery Dennison of Glendale, California. In the embodiment of Fig. 15, the thickness (as measured in the Z direction) of the second substrate 1304 may be 125 microns. The sheet resistance of the resistive material on the back surface of the second substrate 1304 may be 350 kOhm/sq. The layer of adhesive 1308 may be a peel-off adhesive, such as 3M MP467 adhesive. The thickness (as measured in the Z direction) of the layer of adhesive 1308 may be 50 microns. The coverlay 1306 may be made of polyimide, and may have a thickness (as measured in the Z direction) of 12.5 microns. A coverlay adhesive 1502 (e.g., polyethylene with adhesive on either side) may be used to attach the coverlay 1306 to the front surface of the first substrate 1302 atop the metal fingers 1400. The coverlay adhesive 1502 may have a thickness (as measured in the Z direction) of 25 microns. The metal fingers

1400 may be made of copper (e.g., gold-plated copper) and may have a thickness (as measured in the Z direction) of 12.5 microns. The first substrate 1302 may have a thickness (as measured in the Z direction) of 25 microns.

[0132] A pressure sensitive adhesive (PSA) 1504 may be attached to a back surface of the first substrate 1302. The PSA 1504 may be 3M 467MP, and may have a thickness of 50 microns. A PSA liner 1506 may be
5 disposed over the PSA 1504, and may be peeled off before attaching the FSR 1300 to a planar surface (e.g., to a planar surface of a structure mounted inside of a controller body 110).

[0133] At a connector portion of the FSR 1300, a stiffener polyimide 1508 may be attached to the back surface of the first substrate 1302 using a stiffener adhesive 1510. The stiffener polyimide 1508 may have a thickness (as measured in the Z direction) of 137.5 microns, and may create a stiffer connector portion of the FSR
10 1300 for added durability of the connector portion. The thickness (as measured in the Z direction) of the stiffener adhesive may be 25 microns.

[0134] The embodiment of Fig. 15 may represent a FSR 1300 that is suitable for mounting on a planar surface of a structure that is mounted within a handle 112/612 of a controller 100/600 for an electronic system (e.g., a VR system), as disclosed herein. It is to be appreciated that other thickness values, sheet resistance
15 values, and/or materials than those specified with reference to Fig. 15 may be utilized, such as when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these values and materials are to be understood as non-limiting.

[0135] Fig. 16 depicts example layers of a FSR 1300, according to another embodiment of the present disclosure. Fig. 16 is not to scale. Rather, Fig. 16 is presented to illustrate example layers of material, and is not
20 meant to represent an actual cross-sectional view of the FSR 1300. The FSR 1300 shown in Fig. 16 with respect to the first substrate 1302 and the layers above (i.e., in the positive Z direction) the first substrate 1302 may be have a similar construction to the FSR 1300 shown in Fig. 15. Fig. 16 differs from Fig. 15 in the layers below (i.e., in the negative Z direction) the first substrate 1302. Thus, for purposes of brevity, the first substrate 1302 and the layers above (i.e., in the positive Z direction) the first substrate 1302 in Fig. 16 will not be described
25 again, as reference can be made to the description of Fig. 15 for these layers in Fig. 16.

[0136] In the embodiment of Fig. 16, a stiffener 1600 may be attached to the back surface of the first substrate 1302 underneath the main body portion of the FSR 1300 using a stiffener adhesive 1510. The thickness (as measured in the Z direction) of the stiffener adhesive may be 25 microns, as is the case in the embodiment of Fig. 15, but the stiffener 1600 is located underneath the main body portion of the FSR 1300, whereas the stiffener
30 polyimide 1508 is located underneath a connector portion of the FSR 1300. Furthermore, the stiffener 1600 may be a FR4 stiffener having a thickness (as measured in the Z direction) of 530 microns, which is thicker than the stiffener polyimide 1508 of the embodiment of Fig. 15. A pulltab 1602 may be attached to the back surface of the stiffener 1600 using a layer of adhesive 1604. The layer of adhesive 1604 may be a pulltab adhesive, such as 3M MP467 adhesive. The thickness (as measured in the Z direction) of the layer of adhesive 1604 may be 50
35 microns.

[0137] The embodiment of Fig. 16 may represent a FSR 1300 that is suitable for mounting on a planar surface of a structure that is mounted underneath a thumb-operated control 116 of a controller 100/600 for an electronic system (e.g., a VR system), as disclosed herein. It is to be appreciated that other thickness values, sheet resistance values, and/or materials than those specified with reference to Fig. 16 may be utilized, such as
40 when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these values

and materials are to be understood as non-limiting.

[0138] Fig. 17 depicts example layers of a FSR 1300, according to another embodiment of the present disclosure. Fig. 17 is not to scale. Rather, Fig. 17 is presented to illustrate example layers of material, and is not meant to represent an actual cross-sectional view of the FSR 1300. Some of the layers of the FSR 1300 shown in Fig. 17 may have a similar construction to the FSR 1300 shown in Fig. 15. Fig. 17 differs from Fig. 15 several aspects, however.

[0139] In the embodiment of Fig. 17, the thickness (as measured in the Z direction) of the second substrate 1304 may be 127 microns. The layer of adhesive 1308 may be a peel-off adhesive, such as 3M 468MP adhesive. For a FSR 1300 that can withstand high temperatures of a reflow oven, the layer of adhesive 1308 may be a peel-off adhesive, such as 3M 9085, or 3M 9082. The thickness (as measured in the Z direction) of the layer of adhesive 1308 may be 125 microns. In some cases, the thickness of the layer of adhesive 1308 may be 50 microns. In addition, the metal fingers 1400 may be made of RA copper. In addition, a conductive material 1700 may be disposed on a back surface of the first substrate 1302. The conductive material 1700 may be HA copper or RA copper having a thickness (as measured in the Z direction) of 12.5 microns. An additional coverlay 1702 may be deposited on the conductive material 1700. This additional coverlay 1702 may be made of polyimide, and may be attached to the conductive material 1700 using a coverlay adhesive 1704. The thickness (as measured in the Z direction) of the additional coverlay 1702 may be 12.5 microns, and a thickness (as measured in the Z direction) of the coverlay adhesive 1704 may be 25 microns. A layer of adhesive 1706 may be disposed on the coverlay 1702. The layer of adhesive 1706 may be a peel-off adhesive, such as 3M 467MP adhesive, at a thickness (as measured in the Z direction) of 60 microns. For a FSR 1300 that can withstand high temperatures of a reflow oven, the layer of adhesive 1706 may be a peel-off adhesive, such as 3M 9085, or 3M 9082.

[0140] The embodiment of Fig. 17 may represent a FSR 1300 that is suitable for mounting on a planar surface of a structure that is mounted within a controller body 110 of a non-VR controller. It is to be appreciated that other thickness values, sheet resistance values, and/or materials than those specified with reference to Fig. 17 may be utilized, such as when the FSR 1300 is used in other applications, such as non-controller based applications. As such, these values and materials are to be understood as non-limiting.

[0141] Figs. 18A-D depict a FSR 1800, according to another embodiment of the present disclosure. The FSR 1800 may have component layers that are similar to those described with reference to the FSR 1300, such as a first substrate 1802 made of polyimide, and a second substrate 1804 that is flexible and that has a resistive material on its back surface. One or more spacer layers (e.g., a coverlay 1806 and a layer of adhesive 1808) may be interposed between the first substrate 1802 and the second substrate 1804.

[0142] A portion of the first substrate 1802 of the FSR 1800 in Figs. 18B and 18C is wrapped around the second substrate 1804 and is also disposed on a front surface of the second substrate 1804. Fig. 18A is labeled "Before Fold" and depicts the FSR 1800 before the portion of the first substrate 1802 is wrapped around the second substrate 1804. In Fig. 18A, the FSR 1800 includes a first body portion 1812(1) (sometimes referred to as the "lower balloon" 1812(1)) and a second body portion 1812(2) (sometimes referred to as the "upper balloon" 1812(2)). The lower balloon 1812(1) is connected to the upper balloon 1812(2) by a folding neck 1814 at a first end of the lower balloon 1812(1). A soldering pigtail 1816 extends from a second end of the lower balloon 1812(1) and soldering pads 1818 are on the terminal end of the soldering pigtail 1816. An actuator 1810 in the form of a tact switch is disposed on the upper balloon 1812(2) such that the actuator 1810 ends up being the front

or top layer of the FSR 1800 after the folding operation, as shown in Figs. 18B and 18C. Thus, the portion of the first substrate 1802 of the FSR 1800 that is wrapped around the second substrate 1804 is the upper balloon 1812(2).

[0143] A cross section of the FSR 1800, after the folding operation, is shown in Fig. 18C to depict example layers of the FSR 1800. Some of the layers shown in Fig. 18C are described in more detail with reference to Fig. 18D. In this embodiment of Fig. 18C, a force, F , can be applied to the actuator 1810 (e.g., tact switch), causing a variable resistance of the FSR 1800 that is converted into a variable digitized value. The use of a tact switch for the actuator 1810 (e.g., a switch that toggled to a different binary state under the application of a predefined amount of force, F) creates a dual-stage FSR 1800 that first “clicks” when the tact switch 1810 is actuated, and then the FSR 1800 can output variable resistance as increased force, F , is applied. This can serve to calibrate the FSR 1800 on individual actuations of the FSR 1800 by assuming that the tact switch 1810 actuates at a same amount of force, F , each time it is depressed. That is, the FSR 1800 can reset to the known amount of force, F , associated with actuation of the tact switch 1810 in response to detecting an actuation of the tact switch 1810. This can mitigate inherent inaccuracy of the FSR 1800.

[0144] As shown in Figs. 18C and 18D, the FSR 1800 includes a first substrate 1802 made of polyimide having a thickness (as measured in the Z direction) of 25 microns. A conductive material (e.g., the metal fingers 1820 made of HA copper (e.g., gold-plated copper) shown in Fig. 18D) having a thickness (as measured in the Z direction) of 12.5 microns may be disposed on the front surface of the first substrate 1802 at the lower balloon 1812(1) such that the conductive material is underneath the resistive material on the second substrate 1804. A coverlay adhesive 1822 may be used to attach the coverlay 1806 to the front surface of the first substrate 1802 atop the metal fingers 1820. The coverlay adhesive 1822 may have a thickness (as measured in the Z direction) of 25 microns. The coverlay 1806 may be made of polyimide, and may have a thickness (as measured in the Z direction) of 12.5 microns. The layer of adhesive 1808 disposed on the coverlay 1806 may be a peel-off adhesive, such as 3M MP467 adhesive. The thickness (as measured in the Z direction) of the layer of adhesive 1808 may be 60 microns. The thickness (as measured in the Z direction) of the second substrate 1804 may be 127 microns. The sheet resistance of the resistive material on the back surface of the second substrate 1804 may be 350 kOhm/sq. A layer of adhesive 1824 may be used to attach the upper balloon 1812(2) to the lower balloon 1812(1) when the upper balloon 1812(2) is folded over the lower balloon 1812(1) at the folding neck 1814. The layer of adhesive 1824 can be 125 microns in thickness (as measured in the Z direction). A suitable adhesive for the layer of adhesive 1824 is 3M 468MP. The layer of adhesive 1824 may also be C-shaped.

[0145] On the upper balloon 1812(2) of the FSR 1800, a first stiffener polyimide 1834 may be attached to the front surface of the first substrate 1802 (before folding) using a stiffener adhesive 1836. The first stiffener polyimide 1834 may have a thickness (as measured in the Z direction) of 75 microns. The thickness (as measured in the Z direction) of the stiffener adhesive may be 25 microns. In addition, on the upper balloon 1812(2) of the FSR 1800, a second stiffener polyimide 1838 may be attached to the front surface of the first stiffener polyimide 1834 (before folding) using a layer of adhesive 1840. The second stiffener polyimide 1838 may have a thickness (as measured in the Z direction) of 75 microns. The thickness (as measured in the Z direction) of the layer of adhesive may be 125 microns. When the upper balloon 1812(2) is folded over the lower balloon 1812(1) at the folding neck 1814, the second stiffener polyimide 1838 comes into contact with the second substrate 1804, as shown in Fig. 18C, and the layer of adhesive 1824 adheres the two main body portions 1812(1) and 1812(2) of

the FSR 1800 in a stacked relationship after the folding operation. It is to be appreciated that other thickness values, sheet resistance values, and/or materials than those specified with reference to Fig. 18D may be utilized, such as when the FSR 1800 is used in other applications, such as non-controller based applications. As such, these values and materials are to be understood as non-limiting

[0146] In addition, as shown in Fig. 18D, a conductive material 1826 may be disposed on a back surface of the first substrate 1802. The conductive material 1826 may be HA copper having a thickness (as measured in the Z direction) of 12.5 microns. An additional coverlay 1828 may be deposited on the conductive material 1826. This additional coverlay 1828 may be made of polyimide, and may be attached to the conductive material 1826 using a coverlay adhesive 1830. The thickness (as measured in the Z direction) of the additional coverlay 1828 may be 12.5 microns, and a thickness (as measured in the Z direction) of the coverlay adhesive 1830 may be 25 microns. The additional coverlay 1828 and the coverlay adhesive 1830 may span the soldering pigtail 1816, the lower balloon 1812(1), the folding neck 1814, and a portion of the upper balloon 1812(2), leaving a footprint (or space) for the actuator 1810 ("Button Footprint" in Fig. 18D). A layer of adhesive 1832 may be disposed on the additional coverlay 1828. The layer of adhesive 1832 may be a peel-off adhesive, such as 3M 468MP adhesive, at a thickness (as measured in the Z direction) of 125 microns. The layer of adhesive 1832 may span the soldering pigtail 1816 and the lower balloon 1812(1).

[0147] Although the example FSR 1300/1800 is shown as having a generally circular shape, it is to be appreciated that the FSR 1300/1800 can be constructed in layers of different cross-sectional shapes, such as square, rectangular, etc. The FSR 1300/1800 can be larger or smaller in overall size than the examples described herein, depending on the particular application. Furthermore, it is to be appreciated that arrays of FSRs can be implemented by connecting multiple FSRs 1300/1800 together. In such an array, the layers of FSR material may be constructed in a long strip of material.

[0148] Fig. 19 is a flow diagram of an example process 1900 for manufacturing a FSR, such as the FSR 1300 or the FSR 1800 disclosed herein. The processes described herein are illustrated as a collection of blocks in a logical flow graph, which represent a sequence of operations. The order in which the operations are described is not intended to be construed as a limitation, and any number of the described blocks can be combined in any order and/or in parallel to implement the processes.

[0149] At 1902, a first substrate 1302 made of polyimide may be formed with copper clad layer disposed on a front surface of the first substrate 1302.

[0150] At 1904, the copper clad layer may be etched to form a plurality of interdigitated copper fingers (i.e., an example of the metal fingers 1400) on the front surface of the first substrate 1302. The etching at block 1904 may include removing strips of copper material having a width of 0.2 mm to create a distance of 0.2 mm between pairs of adjacent copper fingers among the plurality of interdigitated copper fingers. The spacing between sequential strips of removed copper material may also be kept at 0.2 mm to provide copper fingers having a width of 0.2 mm.

[0151] At 1906, a layer of gold plating may be deposited onto the plurality of interdigitated copper fingers to create gold-plated fingers. This gold plating may be ENIG.

[0152] At 1908, one or more spacer layers may be provided atop the first substrate 1302 at a periphery of the first substrate 1302, thereby leaving a portion of the gold-plated fingers uncovered by the one or more spacer layers. As shown by the sub-blocks 1910 and 1912, multiple spacer layers may be provided in two operations.

[0153] At 1910, a coverlay 1306 (e.g., made of polyimide) may be deposited on the first substrate 1302 at a periphery of the first substrate. The coverlay 1306 may cover a peripheral portion of the gold-plated fingers, wherein a remaining portion of the gold-plated fingers is left uncovered by the coverlay 1306.

[0154] At 1912, a layer of adhesive 1308 may be deposited on the coverlay 1306 such that the remaining portion of the gold-plated fingers is left uncovered by the layer of adhesive 1308. Furthermore, the operation at block 1912 may include leaving a section of the coverlay 1306 uncovered by the layer of adhesive 1308 to create an air gap that allows air to ingress or egress from a space between the first substrate 1302 and the second substrate 1304.

[0155] At 1914, a second substrate 1304 may be provided atop the first substrate 1302 such that a center portion of the second substrate 1304 is suspended over the first substrate 1302 by the one or more spacer layers interposed between the first substrate 1302 and the second substrate 1304. This second substrate 1304 is flexible and has resistive material disposed on a back surface of the second substrate 1304.

[0156] At 1916, in order to construct the FSR 1800, an extended portion of the first substrate 1802 may be wrapped around the second substrate 1804 and attached to the front surface of the second substrate 1804, where the extended portion of the first substrate 1802 is to be interposed between a to-be-attached actuator 1810 and the second substrate 1804. As shown by the dotted outline of block 1916, this operation is performed to construct the FSR 1800, but may be omitted when constructing the FSR 1300.

[0157] At 1918, an actuator 1310 may be provided atop the second substrate 1304, such as by attaching the actuator 1310 to a front surface of the second substrate 1304 to construct the FSR 1300, or by attaching the actuator 1810 (e.g., a tact switch), to the first substrate 1802 that is interposed between the first substrate second substrate 1804 and the actuator 1810.

[0158] The FSR 1300/1800 disclosed herein may be mounted on a planar surface of a structure within a handheld controller, such as the controller 100/600 disclosed herein, and this structure can be positioned at any suitable location within the controller body 110 in order to measure a resistance value that corresponds to an amount of force applied to an outer surface of the controller body 110 (e.g., a force applied by a finger pressing upon a control, a force applied by a hand squeezing the handle 112/612. With specific reference to Figs. 9A and 9B, the FSR 1300/1800 can be mounted on a planar surface of the PCB 920, which itself may be mounted within the tubular housing 612a, 612b of the handle 612. In this configuration, the plunger 924 may interface with the actuator 1310/1810 of the FSR 1300/1800, which may allow for conveying a compressive force from the plunger 924 to the actuator 1310/1810. Other configurations are possible, however, where the plunger 924 is omitted, and the actuator 1310/1810 interfaces with a portion of the tubular housing 612a, 612b of the handle 612. With specific reference to Fig. 1, a FSR 1300/1800 can be mounted on a planar surface of a structure within a head (between the handle 112 and a distal end 111). The structure mounted within a head may be mounted underneath one or more of the thumb-operated controls 114, 115, 116. For example, the FSR 1300/1800 can be positioned underneath the thumb-operated control 116 (e.g., a track pad). Accordingly, when a user's thumb presses upon the thumb-operated control 116 during operation while the controller 100, the FSR 1300/1800 positioned underneath the thumb-operated control 116 may be configured to measure a resistance value that corresponds to an amount of force applied to the thumb-operated control 116 by the user's thumb. It is to be appreciated that multiple FSRs 1300/1800 can be disposed within the controller body 110 of the controller, such as one or more FSRs 1300/1800 mounted within the handle 112/612 and one or more FSRs 1300/1800 mounted underneath one or more corresponding controls 114, 115,

116 on the head of the controller body 110.

[0159] The FSR 1300/1800 disclosed herein may enable variable analog inputs when implemented in a controller 100/600. For instance, squeezing the handle 112/612 or pressing upon the thumb-operated control(s) (e.g., 116) with varying amount of force may cause resistance of the FSR 1300/1800 to vary with the applied force, and the resistance can be converted to a varying digitized value that represents the FSR input for controlling a game mechanic.

[0160] Fig. 20 illustrates an example user interface (UI) 2000 that may be used to configure a FSR-based input mechanism of a handheld controller, such as the controller 100/600, for an electronic system to operate in different modes. The UI 2000 may be output on a display of an electronic system, such as a head mounted display (HMD), or any other type of display used with a personal computer (PC) or a game console. The UI 2000 includes an “Activation Type” drop down menu 2002. The “Activation Type” drop down menu 2002 may be used to select a “Soft Press” type of activation for a FSR-based input mechanism (e.g., the thumb-operated control 116, the handle 112/612, etc.). Here, “Soft Press” means “software press,” which allows the controller 100/600 and/or the electronic system with which the controller 100/600 is associated to determine, using logic, when to register a FSR-based input event based on the analog input of the FSR 1300/1800 (e.g., the FSR resistance, which corresponds to the applied force on the FSR 1300/1800, and which is converted to a digitized FSR input value), and also based on the additional configuration settings that will be discussed shortly. In other words, a resistance value can be measured by the FSR 1300/1800, which can be converted into a digitized FSR input value. If this digitized FSR input value meets a criterion specified by the configuration settings for the “Soft Press,” a FSR-based input event can be registered.

[0161] The UI 2000 may further include a “Binding” drop down menu 2004, which may be used to select a PC-based input control to bind to the corresponding FSR-based input mechanism on the controller 100/600. Here, the binding is selected as the Left Mouse button, but it is to be appreciated that the binding can be selected as other PC-based input controls. The binding can also be analog. For example, for a racing game, the FSR 1300/1800 can be used for the gas pedal (e.g., the harder the user presses upon a FSR-based control mechanism, the faster the racing vehicle goes in the game).

[0162] The UI 2000 may further include a “Soft Press Style” drop down menu 2006, which may be used to select one of various styles of a Soft Press. A “Simple Threshold” style means that a FSR input event occurs when the digitized FSR input value meets or exceeds a threshold value. Because the digitized FSR input value corresponds to a particular resistance value measured by the FSR, which, in turn, corresponds to a particular amount of force applied to the FSR 1300/1800, one can also think of this style of Soft Press as registering a FSR input event when the resistance value measured by the FSR meets a threshold resistance value, and/or when the applied amount of force meets a threshold amount of force. For example, if the handle 112/612 of the controller 100/600 includes a FSR 1300/1800, the handle 112/612 can be squeezed until a threshold amount of force is reached, and, in response, a FSR input event is registered as a “Soft Press.” The force required to “unpress” may be a fraction of the threshold value for debounce purposes and/or to mimic a tact switch with a physical snap ratio. The “Simple Threshold” style may therefor replace a conventional mechanical switch. The UI 200 shows that a configurable Soft Press Threshold 2008(1) can be adjusted by the user to increase or decrease a threshold value that is compared to digitized FSR input values to determine whether to register a FSR input event. The user can adjust the Soft Press Threshold 2008(1) lower (e.g., by moving the slider to the left) in order to reduce hand

fatigue relating to actuation of the FSR-based input mechanism. The user can adjust the Soft Press Threshold 2008(1) higher (e.g., by moving the slider to the right) in order to reduce the instances in which an accidental input is registered by the FSR-based input mechanism. In some cases, the Soft Press Threshold 2008(1) can be set to a default threshold value for a particular game (e.g., a lower default threshold value for a shooting game, a higher default threshold value for an exploration game, etc.).

[0163] A “Hair Trigger” style may set a baseline threshold value, and once a digitized FSR input value associated with the FSR 1300/1800 meets or exceeds the baseline threshold value, the binding is activated (i.e., a FSR input event is registered, akin to a press-and-hold button actuation). Thereafter, any subsequent decrease in force deactivates the binding (i.e., the FSR input event is “unregistered,” akin to a user letting go of a button), and any increase in force after deactivating the binding operates to activate the binding again. There may be some debounce in the “Hair Trigger” style of Soft Press. Turning briefly to Fig. 21, an example of the “Hair Trigger” logic is shown on a Force vs. Time graph 2100. The Force axis may represent digitized FSR input values ranging from zero to any suitable maximum value, which corresponds to a range of resistance values that are measurable by the FSR 1300/1800. As shown in Fig. 21, as the digitized FSR input value increases (e.g., the user presses harder and harder on the FSR-based input mechanism), the digitized FSR input value eventually crosses the baseline threshold value 2102, and, in response, the binding is activated (i.e., a FSR input event is registered akin to a press-and-hold type of user input), and thereafter, the binding is deactivated in response to a decrease in the digitized FSR input value (e.g., the user “lets up” slightly on the FSR-based input mechanism). If the user presses harder upon the FSR-based input mechanism, the binding may be activated again, and so on and so forth, as long as the force remains at a value that is greater than the baseline threshold value 2102.

[0164] With reference again to Fig. 20, a “Hip Fire” style of Soft Press may be selected in three different sub-styles (e.g., Aggressive, Normal, and Relaxed). The “Hip Fire” style may be similar to the “Simple Threshold” style of Soft Press, except that the “Hip Fire” style utilizes a time delay so that, in a configuration with multiple levels of bindings, the time delay can be used to ignore lower FSR input values if a higher threshold value is reached quickly enough. The amount of time delay varies between the different sub-styles (e.g., Aggressive, Normal, and Relaxed). Turning briefly to Fig. 22, an example of the “Hip Fire” logic is shown on a Force vs. time graph 2200. Again, the Force axis may represent a range of digitized FSR input values from zero to any suitable maximum value, which corresponds to a range of resistance values that are measurable by the FSR 1300/1800. As shown in Fig. 22, assume that A1 2202 corresponds to a first threshold value that corresponds to a first action, and A2 2204 corresponds to a second threshold value that corresponds to a second action. A time delay, t , can be set based on whether the Hip Fire style is the Aggressive type, the Normal type, or the Relaxed type. In the “Fast” curve shown in Fig. 22, the FSR input values reach A1 2202 quickly, which triggers the time delay to start running. The FSR input values then reach A2 2204 before the time delay has lapsed, which causes the logic to ignore A1 2202 and to register a FSR input event exclusively for the second action that corresponds to A2 2204. In the “Slow” curve shown in Fig. 22, the FSR input values reach A1 2202, and the time delay is started. However, because the FSR input values do not increase fast enough to reach A2 2204 before the time delay lapses, the logic registers a FSR input event for the first action that corresponds to A1 2202, and, thereafter, the FSR input values eventually reach A2 2204, and the logic registers an additional FSR input event for the second action that corresponds to A2 2204. The time delay, t , may be specified in milliseconds, and is configurable.

[0165] With reference again to Fig. 20, an additional Soft Press Threshold 2008(2) may be usable, for example, to set a multi-level threshold, such as the thresholds for the “Hip Fire” style of Soft Press. The different styles of Soft Press for FSR-based input can be used to enable a number of different game-related, analog inputs by virtue of the user squeezing or pressing a FSR-based input mechanism with varying force. For example, a VR game can allow a user to crush a rock or squeeze a balloon by squeezing the handle 112/612 of the controller body 110 with increasing force. As another example, a shooting-based game may allow the user to toggle between different types of weapons by pressing a thumb-operated control 116 with different levels of applied force.

[0166] Fig. 23 depicts the controller 100 of Fig. 1 having various sensors disposed within the controller body 110. For example, a first FSR 1300(1) may be mounted underneath a control that is configured to be pressed, such as the thumb-operated control 116 that is included on a head 113 of the controller body 110. A second FSR 1300(2) may be mounted within the handle 112 of the controller body 110, along with an array of proximity sensors 800. It is to be appreciated that one or the other FSR 1300(1) or 1300(2) may be provided within the controller 100, or both FSRs 1300(1) and 1300(2) may be provided within the controller 100. In addition, or alternatively to array of proximity sensors 800, one or more touch sensors 2300 (e.g., touch sensors 2300(1)-(3)) may be associated with one or more controls that are configured to be pressed, such as the thumb-operated control 114, the thumb-operated control 115, and/or the thumb-operated control 116, and/or a finger-operated control (e.g., a trigger 609). The touch sensor(s) 2300 may be configured to provide touch data indicative of an object (e.g., a finger, a thumb, etc.) contacting an associated control (e.g., one or more of the thumb-operated controls 114-116). In an example, the touch sensor(s) 2300 comprises a capacitive sensor (or array of capacitive sensors) that is mounted within the head 113 of the controller body 110 (e.g., adhered or otherwise attached to a back surface of the outer housing and underneath the controls 114-116, attached to a structure, such as a PCB, within the head 113, etc.). In other instances, the touch sensor(s) 2300 can be based on other touch-sensing technologies, such as an infrared or acoustic touch sensor. Meanwhile, the array of proximity sensors 800 that are spatially distributed on the handle 112 may be configured to provide proximity data indicative of a hand grasping the handle 112. The proximity sensors 800 can also use any suitable technology for sensing contact and/or proximity of the hand on/to the handle 112, as disclosed herein. The FSRs 1300 are configured to provide force data indicative of an amount of force of a press of a control (e.g., a press of the control 116), or a squeeze of the handle 112. Sets of the various sensors shown in Fig. 23 may be connected by a flex circuit. For example, the touch sensors 2300 and the FSR 1300(1) in the head 113 may be connected by a common flex circuit. The polyimide substrate of the FSR 1300 disclosed herein allows for this type of direct soldering of the FSR output terminals to a flex circuit.

[0167] The processes described herein are illustrated as a collection of blocks in a logical flow graph, which represent a sequence of operations that can be implemented in hardware, software, or a combination thereof. In the context of software, the blocks represent computer-executable instructions that, when executed by one or more processors, perform the recited operations. Generally, computer-executable instructions include routines, programs, objects, components, data structures, and the like that perform particular functions or implement particular abstract data types. The order in which the operations are described is not intended to be construed as a limitation, and any number of the described blocks can be combined in any order and/or in parallel to implement the processes.

[0168] Fig. 24 is a flow diagram of an example process 2400 for re-calibrating a FSR 1300/1800 of a handheld controller 100/600 based on touch data provided by a touch sensor.

[0169] At 2402, logic of the handheld controller 100/600 may determine, based at least in part on touch data provided by a touch sensor, that an object (e.g., a finger, a thumb, etc.) has come into contact with at least one control of the handheld controller. The at least one control may be included on the controller body 110 of the controller 100/600 and may be configured to be pressed. For instance, the control may be the thumb-operated control 116 included on the head 113 of the controller body 110. In this embodiment, the touch sensor can be one of the touch sensors 2300. Alternatively, the control may be the handle 112 of the controller body 110. In this embodiment, the touch sensor can be the array of proximity sensors 800.

[0170] At 2404, the logic may determine, based at least in part on force data provided by a FSR 1300/1800 at a time at which the object has come into contact with the at least one control, a resistance value measured by the FSR 1300/1800.

[0171] At 2406, the logic may correlate the resistance value with a digitized FSR input value of zero. In other words, the sensed resistance when the object comes into contact with the at least one control can be taken as a force input of zero, meaning that any increase in force applied to the FSR 1300/1800 after that point is correlated with a positive FSR input value. Thus, the process 2400 represents a sensor fusion algorithm that can help mitigate any inherent inaccuracy of the FSR 1300/1800 – which might measure some resistance even when an object is not pressing upon the control – through recalibration on the detection of a touch of the control.

[0172] Fig. 25 is a flow diagram of an example process 2500 for ignoring spurious input at a FSR 1300/1800 of a handheld controller 100/600 based on touch data provided by a touch sensor for an adjacent control.

[0173] At 2502, logic of a handheld controller 100/600 may determine, based at least in part on force data provided by a FSR 1300/1800 associated with a first control (e.g., thumb-operated control 116) of the handheld controller, a resistance value measured by the FSR 1300/1800.

[0174] At 2504, the logic may convert the resistance value to a digitized FSR input value.

[0175] At 2506, the logic may determine whether the digitized FSR input value meets or exceeds a threshold value that is to be met in order to register a FSR input event for the first control. If the threshold is not met at 2506, the process 2500 follows the “NO” route from block 2506 to block 2502 to await additional force data. If the threshold is met at 2506, the process 2500 follows the “YES” route from block 2506 to block 2508.

[0176] At 2508, the logic may determine whether an object (e.g., a finger, a thumb, etc.) is in contact with an adjacent, second control based at least in part on touch data provided by a touch sensor 2300 associated with a second control adjacent to the first control (e.g., thumb-operated control 114 or 115) – the touch data provided at a time at which the FSR resistance value is measured by the FSR 1300/1800. If the object is not in contact with the adjacent, second control, the process 2500 follows the “NO” route from block 2508 to block 2510, where the logic registers a FSR input event for the first control (e.g., by activating a binding for the first control). If the object is in contact with the adjacent, second control, the process 2500 follows the “YES” route from block 2508 to block 2512.

[0177] At 2512, the logic may refrain from registering the FSR input event for the first control based at least in part on determining that the object is in contact with the second control. Thus, the process 2500 represents a sensor fusion algorithm that can be used to ignore spurious inputs at the FSR 1300/1800 based on a press of an adjacent control on the handheld controller.

[0178] Fig. 26 is a flow diagram of an example process 2600 for adjusting a FSR input threshold for a FSR 1300/1800 based on a hand size detected by an array of proximity sensors 800 in the handle 112/612 of the handheld controller 100/600.

[0179] At 2602, logic of a handheld controller 100/600 may determine, based at least in part on proximity data provided by an array of proximity sensors 800 that are spatially distributed on a handle of the controller 100/600, a size of a hand that is grasping the handle 112/612. The size of the hand may be determined from among a plurality of predefined hand sizes (e.g., small and large, or small, medium, and large, etc.).

[0180] At 2604, the logic may adjust, based at least in part on the size of the hand determined at block 2602, a threshold value to an adjusted threshold value that is to be met in order to register a FSR input event for the handle 112/612. This adjusted threshold value corresponds to a particular amount of force with which the handle 112/612 can be squeezed. For example, an amount of force corresponds to a measured resistance of the FSR 1300/1800 in the handle 112/612, and that resistance may correspond to a digitized FSR input value. When the user squeezes the handle, if the digitized FSR input value meets or exceeds the adjusted threshold value, a FSR input event can be registered. Thus, the threshold value may be adjusted to a lower value for a user with a smaller hand, while the threshold value may be adjusted to a greater value for a user with a larger hand, as detected by the array of proximity sensors 800 at block 2602. In some cases, a default threshold value may be configured for the controller 100/600 prior to detection of the hand size at block 2602, and the adjustment at block 2604 may be to increase or decrease the threshold value with respect to the default value.

[0181] As shown by the sub-blocks in Fig. 26, the process 2600 may involve more detailed operations. For example, the determining of the hand size at block 2602 may include sub-blocks 2606 and 2608.

[0182] At 2606, the logic may determine a number of proximity sensors of the array of proximity sensors 800 that provided the proximity data. For example, a small hand may only span a small subset of the proximity sensors in the array of proximity sensors 800, and the remaining proximity sensors that do not detect the small-sized hand may not provide the aforementioned proximity data. By contrast, a large hand may span the entirety of the array of proximity sensors 800, and, in this case, all (or at least a number above a threshold number) of the proximity sensors 800 may provide the proximity data.

[0183] At 2608, the logic may determine the size of the hand based at least in part on the number of proximity sensors (of the array 800) that provided the proximity data.

[0184] Additionally, as shown by sub-blocks 2610 and 2612, the adjustment of the threshold value at block 2604 may include adjusting the threshold value for one or more FSRs of the controller 100/600.

[0185] For example, at 2610, the logic may adjust a first threshold value (associated with a first FSR 1300(1)) that is to be met in order to register a FSR input event for the control 116. At 2612, the logic may, additionally or alternatively, adjust a second threshold value (associated with a second FSR 1300(2)) that is to be met in order to register a FSR input event for the handle 112/612.

[0186] Fig. 27 is a flow diagram of an example process 2700 for activating and deactivating a binding for a control of a handheld controller based on FSR input values. As shown by the off-page reference “A” in Fig. 27, the process 2700 may continue from any of the processes 2400, 2500, or 2600, but it does not have to.

[0187] At 2702, logic of a handheld controller 100/600 may determine, based at least in part on force data provided by a FSR 1300/1800 of the controller 100/600, at a first time, a first digitized FSR input value. This first digitized FSR input value may be converted from a first resistance value measured by the FSR 1300/1800 at

the first time.

[0188] At 2704, the logic may determine whether the first digitized FSR input value meets or exceeds a threshold value that is to be met in order to register a FSR input event (e.g., for binding the control associated with the FSR 1300/1800). If the threshold is not met at 2704, the process 2700 follows the “NO” route from block 2704 to block 2702 where the logic awaits additional force data. If the threshold is met at 2704, the process 2700 follows the “YES” route from block 2704 to block 2706.

[0189] At 2706, the logic may register the FSR input event (e.g., to activate the binding associated with the control associated with the FSR 1300/1800) based at least in part on the first digitized FSR input value meeting or exceeding the threshold value.

[0190] At 2708, the logic may determine, based at least in part on force data provided by the FSR 1300/1800, at a second time after the first time, a second digitized FSR input value. This second digitized FSR input value may be converted from a second resistance value measured by the FSR 1300/1800 at the second time.

[0191] At 2710, the logic may determine whether the second digitized FSR input value is less than the first digitized FSR input value (i.e., whether the FSR input has decreased since a previous measurement by the FSR 1300/1800). If the second digitized FSR input value is less than the first digitized FSR input value, the process 2700 follows the “YES” route from block 2710 to block 2712, where the logic may deactivate the binding for the control associated with the FSR 1300/1800 (which can be thought of as unregistering a previously registered FSR input event that amounts to a press-and-hold input). If the second digitized FSR input value is not less than the first digitized FSR input value at block 2710, the process 2700 follows the “NO” route from block 2710 to block 2708 where the logic awaits additional force data from the FSR 1300/1800. The process 2700 may reflect the FSR detection mode illustrated in Fig. 21 and described above. Thus, the threshold evaluated at block 2704 may correspond to the baseline threshold value 2102 described with reference to Fig. 21.

[0192] Fig. 28 is a flow diagram of an example process 2800 for using a time-delay to determine whether to ignore FSR input for a first of multiple thresholds. As shown by the off-page reference “A” in Fig. 28, the process 2800 may continue from any of the processes 2400, 2500, or 2600, but it does not have to.

[0193] At 2802, logic of a handheld controller 100/600 may determine, based at least in part on force data provided by a FSR 1300/1800 of the controller 100/600, at a first time, a first digitized FSR input value. This first digitized FSR input value may be converted from a first resistance value measured by the FSR 1300/1800 at the first time.

[0194] At 2804, the logic may determine whether the first digitized FSR input value meets or exceeds a first threshold value (e.g., A1 2202 of Fig. 22) that is to be met in order to register a first FSR input event (e.g., for binding the control associated with the FSR 1300/1800). The first FSR input event may be associated with a first action (e.g., a first game mechanic). If the first threshold is not met at 2804, the process 2800 follows the “NO” route from block 2804 to block 2802 where the logic awaits additional force data. If the threshold is met at 2804, the process 2800 follows the “YES” route from block 2804 to block 2806.

[0195] At 2806, the logic may start monitoring a predefined time period (e.g., the time delay, t , in Fig. 22).

[0196] At 2808, the logic may determine, based at least in part on force data provided by the FSR 1300/1800, at a second time after the first time, a second digitized FSR input value. This second digitized FSR input value may be converted from a second resistance value measured by the FSR 1300/1800 at the second time.

[0197] At 2810, the logic may determine whether the second digitized FSR input value meets or exceeds a

second threshold value (e.g., A2 2204 of Fig. 22) that is to be met in order to register a second FSR input event (e.g., for binding the control associated with the FSR 1300/1800). The second FSR input event may be associated with a second action (e.g., a second game mechanic) different from the first action, and the second threshold is greater than the first threshold. If the second threshold is not met at 2810, the process 2800 follows the “NO” route from block 2810 to block 2812 where the logic awaits determines whether the predefined time period has lapsed (e.g., whether the difference between the second time and the first time is less than the predefined time period). If the time period has not yet lapsed at block 2812, the process 2800 iterates by following the “NO” route from block 2812 back to block 2810. If the time period has lapsed at block 2812 and the second threshold has not been met, the process 2800 follows the “YES” route from block 2812 to block 2814 where the logic may register the first FSR input event for the first threshold (e.g., which may be associated with a first action or game mechanic).

[0198] If the second threshold is met at 2810, the process 2800 follows the “YES” route from block 2810 to block 2816, where the logic evaluates the predefined time period.). If the time period has not yet lapsed at block 2816, the process 2800 follows the “NO” route from block 2816 back to block 2818 where the logic refrains from registering the first FSR input event, and registers the second FSR input event associated with the second threshold (e.g., which may be associated with a second action or game mechanic). If the time period has lapsed at block 2816 and the second threshold has been met, the process 2800 follows the “YES” route from block 2816 to block 2820 where the logic may register both the first FSR input event for the first threshold and the second FSR input event for the second threshold. The process 2800 may reflect the FSR detection mode illustrated in Fig. 22 and described above.

[0199] Fig. 29 illustrates example components of a handheld controller, such as the controller 100 of Fig. 1, although the components shown in Fig. 29 can be implemented by the controller 600 as well. As illustrated, the handheld controller includes one or more input/output (I/O) devices 2902, such as the controls described above (e.g., joysticks, trackpads, triggers, etc.), potentially any other type of input or output devices. For example, the I/O devices 2902 may include one or more microphones to receive audio input, such as user voice input. In some implementations, one or more cameras or other types of sensors (e.g., inertial measurement unit (IMU)) may function as input devices to receive gestural input, such as motion of the handheld controller 100. In some embodiments, additional input devices may be provided in the form of a keyboard, keypad, mouse, touch screen, joystick, control buttons and the like. The input device(s) may further include control mechanisms, such as basic volume control button(s) for increasing/decreasing volume, as well as power and reset buttons.

[0200] The output devices, meanwhile, may include a display, a light element (e.g., LED), a vibrator to create haptic sensations, a speaker(s) (e.g., headphones), and/or the like. There may also be a simple light element (e.g., LED) to indicate a state such as, for example, when power is on. While a few examples have been provided, the handheld controller may additionally or alternatively comprise any other type of output device.

[0201] In addition, the handheld controller 100 may include one or more communication interfaces 2904 to facilitate a wireless connection to a network and/or to one or more remote systems (e.g., a host computing device executing an application, a game console, etc.). The communication interfaces 2904 may implement one or more of various wireless technologies, such as Wi-Fi, Bluetooth, radio frequency (RF), and so on. It is to be appreciated that the handheld controller 100 may further include physical ports to facilitate a wired connection to a network, a connected peripheral device, or a plug-in network device that communicates with other wireless

networks.

[0202] In the illustrated implementation, the handheld controller further includes one or more processors 2906 and computer-readable media 2908. In some implementations, the processor(s) 2906 may include a central processing unit (CPU), a graphics processing unit (GPU), both CPU and GPU, a microprocessor, a digital signal processor or other processing units or components known in the art. Alternatively, or in addition, the functionally described herein can be performed, at least in part, by one or more hardware logic components. For example, and without limitation, illustrative types of hardware logic components that can be used include field-programmable gate arrays (FPGAs), application-specific integrated circuits (ASICs), application-specific standard products (ASSPs), system-on-a-chip systems (SOCs), complex programmable logic devices (CPLDs), etc. Additionally, each of the processor(s) 2906 may possess its own local memory, which also may store program modules, program data, and/or one or more operating systems.

[0203] In general, the controller may include logic (e.g., software, hardware, and/or firmware, etc.) that is configured to implement the techniques, functionality, and/or operations described herein. The computer-readable media 2908 may include volatile and nonvolatile memory, removable and non-removable media implemented in any method or technology for storage of information, such as computer-readable instructions, data structures, program modules, or other data. Such memory includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, RAID storage systems, or any other medium which can be used to store the desired information and which can be accessed by a computing device. The computer-readable media 2908 may be implemented as computer-readable storage media ("CRSM"), which may be any available physical media accessible by the processor(s) 2906 to execute instructions stored on the computer-readable media 2908. In one basic implementation, CRSM may include random access memory ("RAM") and Flash memory. In other implementations, CRSM may include, but is not limited to, read-only memory ("ROM"), electrically erasable programmable read-only memory ("EEPROM"), or any other tangible medium which can be used to store the desired information and which can be accessed by the processor(s) 2906.

[0204] Several modules such as instruction, datastores, and so forth may be stored within the computer-readable media 2908 and configured to execute on the processor(s) 2906. A few example functional modules are shown as stored in the computer-readable media 2908 and executed on the processor(s) 2906, although the same functionality may alternatively be implemented in hardware, firmware, or as a system on a chip (SOC).

[0205] An operating system module 2910 may be configured to manage hardware within and coupled to the handheld controller 100 for the benefit of other modules. In addition, the computer-readable media 2908 may store a network-communications module 2912 that enables the handheld controller 100 to communicate, via the communication interfaces 2904, with one or more other devices, such as a personal computing device executing an application (e.g., a game application), a game console, a HMD, a remote server, or the like. The computer-readable media 2908 may further include a game-session database 2914 to store data associated with a game (or other application) executing on the handheld controller or on a computing device to which the handheld controller 100 couples. The computer-readable media 2908 may also include a device-record database 2916 that stores data associated with devices to which the handheld controller 100 couples, such as the personal computing device, game console, HMD, remote server or the like. The computer-readable media 2908 may further store game-

control instructions 2918 that configure the handheld controller 100 to function as a gaming controller, and universal-control instructions 2920 that configure the handheld controller 100 to function as a controller of other, non-gaming devices.

[0206] Fig. 30 illustrates an example series of gestures that may be performed using the handheld controller 100 described herein. As illustrated, the handheld controller may couple (wired or wirelessly) to one or more display devices 3000. The display devices 3000 may be representative of any type of device configured to present, on a display, images such as virtual objects and the like. As illustrated, the display devices 3000 may include a server, a game console, a client computing device, a virtual-reality headset, or any other type of device configured to cause presentation of images on a display. In some instances, a display device may include a display (e.g., in the case of a VR headset), while in other instances the display device may communicate data for presentation on a local display (e.g., in the case of a local game console) or a remote display (e.g., in the case of a server computing device).

[0207] In this example, handheld controller 100 may include some or all of the sensors described above, such as proximity sensors, force sensors, accelerometers, and/or the like. The controller 100 and/or a display device 3000 coupled to the controller 100 may utilize the data collected by these sensors to identify a user intending to pick up and/or release a virtual object being presented on the display. In the illustrated example, the display device 3000 presents a first frame 3002(1) that includes a virtual hand 3004 and a virtual object 3006. Sometime at or after the first frame 3002(1), the controller 100 and/or the connected display device may analyze the sensor data to determine that the user intends to grasp the virtual object. As such, a second frame 3002(2) illustrates the virtual hand 3004 holding the virtual object 3006.

[0208] Again, sometime at or after the second frame 3002(2), the controller and/or the display device 3000 may analyze the sensor data collected by the controller to determine that the user intends to release the item. After storing an indication that a release has been initiated, the sensor data collected thereafter may be analyzed for determine when to execute the release—for example, when to generate a representation on the display of the virtual hand 3004 releasing the virtual object 3006. In this example, a third frame 3002(3) illustrates the virtual hand 3004 releasing the virtual object 3006 based on analysis of the sensor data.

[0209] Fig. 31 illustrates example components of the example display devices 3000 and example components of the example handheld controller 100, which may be used to identify when a virtual object is to be picked up and/or released. As illustrated, the handheld controller 100 includes one or more processors 3100, one or more sensors 3102, and one or more computer-readable media 3104. The sensors may include an accelerometer 3106 configured to generate velocity data corresponding to a current velocity of the controller 100, proximity sensors 3108 configured to determine (e.g., via capacitance readings) a proximity of objects (e.g., fingers or other portions of a hand of a user) to the controller 100, force sensing resistors 3110 configured to determine a force applied to the controller 100, and/or one or more other sensors 3112.

[0210] The computer-readable media, meanwhile, may store or otherwise have access to the sensor data 3114 collected by these sensors 3102. As illustrated, the sensor data 3114 may include force values measured by the force sensing resistors 3110, capacitance values measured by the proximity sensors 3108, velocity data measured by the accelerometer 3106, difference values based on a difference between readings at different times (e.g., velocity differences over time), and/or the like. In addition, the computer-readable media may store or otherwise have access to threshold data 3116. The threshold data 3116 may represent thresholds at which

corresponding actions are to be taken. For example, the threshold data 3116 may include a force threshold, a capacitance threshold, a difference threshold, a time threshold, and/or the like. These thresholds are described further with reference to Figs. 32-35.

[0211] The computer-readable media 3104 may further store an object-interaction module 3118 that is configured to analyze the sensor data 3114, with reference to the threshold data 3116, to determine when to take certain actions with reference to virtual objects being presented on a display. For example, the object-interaction module 3118 may be configured to determine when an object is to be picked up, released, smashed, and/or the like. As illustrated, the object-interaction module 3118 may include an object-grip module 3120 configured to analyze the sensor data 3114 and determine when the sensor data meets the criteria for a virtual object to be held. The object-grip module 3120 may include a force-pinch calculator 3122, a force-grab calculator 3124, a cap-pinch calculator 3126, and a cap-grab calculator 3128. In some instances, the force-pinch calculator 3122 may identify a force-pinch state in response to one or more force sensors generating force values that are over a force threshold and one or more buttons being pressed. For example, if a force sensor located on a top side of the handheld controller beneath where a thumb of the user sits returns a force value greater than a particular threshold (e.g., 1% of a maximum force reading of the sensor, 25% of a maximum force reading, etc.) and if a trigger button selectable by a pointer or other finger of the user is pressed, then the described techniques may identify the force-pinch state. That is, in response to these sensor readings, the force-pinch calculator 3122 may determine that the user is attempting to hold a virtual object via a “pinch”. Meanwhile, the force-grab calculator 3124 may identify a force-grip state in response to one or more force sensors on a controller body of the handheld controller measuring one or more force values greater than a force threshold (e.g., 10% of the maximum force reading of the sensor, 25% of the maximum force reading of the sensor, etc.).

[0212] The cap-pinch calculator 3126, meanwhile, may identify a cap-pinch state in based on one or more proximity values (e.g., capacitive value) measured by one or more of the proximity sensors, such as the values measured by a proximity sensor on a top portion of the controller where a thumb of the user sits and by a proximity sensor on a controller body of the controller where an index or other finger of the user sits. In some instances, the software and/or hardware may average these two readings and compare the average to a threshold. If the average value is greater than the threshold (e.g., 30% of the maximum sensor reading, 60% of the maximum sensor reading, etc.), then a cap-pinch state may be identified. The cap-grab calculator 3128, meanwhile, may identify a cap-grip state based on the proximity values recorded by one or more proximity sensors residing on a controller body of the controller. For example, the proximity values measured by proximity sensors corresponding to a middle, ring, and pinky finger of the user may be averaged and compared to a threshold. If this average is greater than a threshold (e.g., 50% of the maximum sensor reading, 70% of the maximum sensor reading, etc.), then a cap-grip state may be identified. Of course, while a few example hold states are described, other types of hold states may be utilized. In addition, Fig. 33 describes operation of the object-grip module 3120 below in greater detail.

[0213] The object-interaction module 3118 may further include a drop-initiation module 3130 and a drop-object module 3132. After the object-grip module 3120 has determined that a virtual object is being held (and thus causes presentation of the virtual object being held), the drop-initiation module 3130 may analyze the sensor data 3114 to determine when to initiate a release of the virtual object. In some instances, a release of a virtual object may be initiated based on a significant drop in force on the controller 100, based on an increase in velocity of the controller 100, based on the proximity readings indicating a lack of a presence of the hand or a portion of

the hand of the user and/or the like. Fig. 34 describes operation of the drop-initiation module 3122 below in greater detail.

[0214] The drop-object module 3132, meanwhile, may be configured to execute a release of an item after the drop-initiation module 3130 has indicated that the release has been initiated. For example, the drop-object module 3132 may analyze the sensor data to determine when to begin representing the release of the object, such as the object being thrown, dropped, smashed, or the like. In some instances, the drop-object module 3132 may determine to release a virtual object based on the velocity data indicating that the controller has effectively ceased moving or has otherwise dropped in velocity, based on expiration of a predetermined amount of time, or the like. Figs. 35-39 describes operation of the object-drop module 3132 below in greater detail.

[0215] It is noted that while Fig. 31 illustrates the handheld controller 100 as including the components of the object-interaction module 3118, in some instances some or all of these components may additionally or alternatively reside on one or more of the display devices 3000. As such, Fig. 31 illustrates the object-interaction module 3118, the sensor data 3114, and the threshold data 3116 in dashed lines.

[0216] Figs. 32-35 collectively illustrate an example process 3200 that the object-interaction module may implement for causing a display device to present the grasping and releasing of virtual objects. In some instances, the object-interaction module 3118 may perform some or all of the process 3200, potentially in combination with one or more other computing-device components, such as a display driver, sensors, and/or the like. Furthermore, in some instances functionality of the object-interaction module may reside across multiple devices and, thus, the operations may be performed by a combination of devices.

[0217] An operation 3202 represents determining whether a controller frame has been updated. For instance, the operation represents determining whether a display is to be updated based, for example, on interaction of a user with a handheld controller and/or based on other inputs of a currently executing application, such as a gaming application. If the controller frame has not been updated, then the process 3200 simply loops back to await a controller frame update. Upon such an update, the object-interaction module 3118 may determine, at an operation 3204, whether a virtual object is currently being held, released, or neither held nor released. In the latter instances, the process 3200 proceeds to the operations described below with reference to Fig. 33. If the object-interaction module 3118 determines that a virtual object is currently being held, then the process 3200 may proceed to the operations described below with reference to Fig. 34. Finally, if the object-interaction module 3118 determines that a virtual object is being released, then the process 3200 may proceed to the operations described with reference to FIG. 35.

[0218] Fig. 33 includes, at an operation 3302, receiving sensor data from one or more of the sensors described above, such as proximity sensors, force resisting sensors, accelerometers, and/or the like. At an operation 3304, the object-grip module 3120 may determine when the received sensor data meets the criteria for force-pinch-type grab, as discussed above. For instance, the object-grip module 3120 may identify a force-pinch state in response to one or more force sensors generating force values that are over a force threshold and one or more buttons being depressed. For example, if a force sensor located on a top side of the handheld controller beneath where a thumb of the user sits returns a force value greater than a particular threshold (e.g., 1% of a maximum force reading of the sensor, 25% of a maximum force reading, etc.) and if a trigger button selectable by a pointer or other finger of the user is pressed, then the object-grip module 3120 may identify the force-pinch state. That is, in response to these sensor readings, the object-grip module 3120 may determine that the user is attempting to hold a

virtual object via a “pinch”. If the criteria for a force-pinch grab are met, then at an operation 3306 the object-grip module 3120 may store an indication of a force-pinch hold for an object that is closest to the location of the virtual hand. That is, the object-grip module 3120 may determine that a nearest virtual object is to be held by a virtual hand of the user based on the received sensor data. At an operation 3308, the object-grip module 3120 may then cause presentation of a virtual hand of the user holding the closest virtual object via a force-pinch grip (e.g., via the pointer finger and thumb of the user). Thereafter, the process 3300 proceeds to an operation 3310, which includes returning to the operation 3202.

[0219] If, however, the object-grip module 3120 determines that the received sensor data does not meet the criteria for a force-pinch grab type, then at 3312 the object-grip module 3120 may determine when the received sensor data meets the criteria for force-grip-type grab, as discussed above. For example, the object-grip module 3120 may identify a force-grip state in response to one or more force sensors on a controller body of the handheld controller measuring one or more force values greater than a force threshold (e.g., 10% of the maximum force reading of the sensor, 25% of the maximum force reading of the sensor, etc.). If these criteria are met, then at an operation 3314 the object-grip module 3120 may store an indication of a force-grip hold for an object that is closest to the location of the virtual hand. That is, the object-grip module 3120 may determine that a nearest virtual object is to be held by a virtual hand of the user based on the received sensor data. At an operation 3316, the object-grip module 3120 may then cause presentation of a virtual hand of the user holding the closest virtual object via a force-grip grab (e.g., via multiple fingers of the user). Thereafter, the process 3300 proceeds to an operation 3310, which includes returning to the operation 3202.

[0220] If, however, the object-grip module 3120 determines that the received sensor data does not meet the criteria for a force-grasp grab type, then at 3318 the object-grip module 3120 may determine when the received sensor data meets the criteria for cap-pinch grab, as discussed above. For instance, the object-grip module 3120 may identify a cap-pinch state based on one or more proximity values (e.g., capacitive value) measured by one or more of the proximity sensors, such as the values measured by a proximity sensor on a top portion of the controller where a thumb of the user sits and by a proximity sensor on a controller body of the controller where an index or other finger of the user sits. In some instances, the object-grip module 3120 may average these two readings and compare the average to a threshold. If the average value is greater than the threshold (e.g., 30% of the maximum sensor reading, 60% of the maximum sensor reading, etc.), then a cap-pinch state may be identified. If so, then at an operation 3320 the object-grip module 3120 may store an indication of a cap-pinch hold for an object that is closest to the location of the virtual hand. That is, the object-grip module 3120 may determine that a nearest virtual object is to be held by a virtual hand of the user based on the received sensor data. At an operation 3322, the object-grip module 3120 may then cause presentation of a virtual hand of the user holding the closest virtual object via a cap-pinch grab (e.g., between a pointer finger and thumb of the user). Thereafter, the process 3300 proceeds to an operation 3310, which includes returning to the operation 3202.

[0221] If, however, the object-grip module 3120 determines that the received sensor data does not meet the criteria for a cap-pinch grab type, then at 3324 the object-grip module 3120 may determine when the received sensor data meets the criteria for cap-grip-type grab, as discussed above. For instance, the object-grip module 3120 may identify a cap-grip hold based on the proximity values recorded by one or more proximity sensors residing on a controller body of the controller. For example, the proximity values measured by proximity sensors corresponding to a middle, ring, and pinky finger of the user may be averaged and compared to a threshold. If this

average is greater than a threshold (e.g., 50% of the maximum sensor reading, 70% of the maximum sensor reading, etc.), then a cap-grip state may be identified. If so, then at an operation 3326 the object-grip module 3120 may store an indication of a cap-grip hold for an object that is closest to the location of the virtual hand. That is, the object-grip module 3120 may determine that a nearest virtual object is to be held by a virtual hand of the user based on the received sensor data. At an operation 3328, the object-grip module 3120 may then cause presentation of a virtual hand of the user holding the closest virtual object via a cap-grip grab (e.g., via multiple fingers of the user). Thereafter, the process 3300 proceeds to an operation 3310, which includes returning to the operation 3202.

[0222] It is to be appreciated that while FIG. 33 describes one example process 3200(1) for determining whether an object is being held, other processes may be used. For example, while process 3200(1) implements a prioritization where a force-based grabs are analyzed and identified prior to cap-based grabs, in other instances this may be reversed. That is, in other implementations cap-based grabs may be prioritized over force-based grabs, may be intermixed in terms of priority with force-based grabs, or the like. In some instances, the prioritization of grab types may depend on the object being grabbed, the last grabbed object, or the like. For example, a first type of object may be associated with a prioritization that first checks for force grabs rather than cap grabs, while the opposite may be true for a second type of object. In addition, or in the alternative, a first type of object may be associated with prioritization that first checks for a pinch grab (force or cap) before checking for a grip grab, while the opposite may be true for a second type of object. For example, if a user attempts to grab an object of a first size that is less than a threshold (e.g., a ping pong ball), the techniques may first attempt to identify a pinch grab, while if the user attempts to grab an object of a second size that is not less than a threshold (e.g., a grenade), then the techniques may attempt to identify a grip grab.

[0223] In addition, the determination of the “closest” object to a hand of a user may be determined in a number of ways. In some instances, the distance between each object in a virtual environment and a virtual hand (or other implement) of the user may be measured between a first point of the object (e.g., a centroid, a closest point on the surface of the object, etc.) and a particular point of the virtual hand. This latter point may be based on the type of grab the user is attempting to perform, the most recent grab the user performed the type of object the user is attempting to grab, the object that the user most recently held, or the like. For example, if the user is attempting to perform a pinch-type grab (or if the most recent grab of the user was a pinch-type grab), then the reference point of the virtual hand for determining a distance to each object in the environment may be a point between the thumb and index finger of the virtual hand of the user. If, however, the user is attempting to perform a grip-type grab (or if the most recent grab of the user was a grip-type grab), then the reference point of the virtual hand for determining a distance to each object in the environment may be a point corresponding to a centroid of the user’s virtual hand or fist. Of course, while a few examples are described, it is to be appreciated that the “closest object” may be determined in multiple other ways.

[0224] Upon returning to the operation 3202, the process proceeds down the “holding” fork of decision block 3204, given that a virtual object is now being held by a virtual hand of the user. The process 3200 thus proceeds to the Fig. 34, which includes, at 3402, again receiving additional sensor data 3402 from the sensors onboard the controller 100. At 3404, the drop-initiation module 3130 may determine whether the current grab-type is capacitive. If so, then at 3406 the drop-initiation module 3130 may determine whether the sensor data indicates that the capacitive grab-type now meets the criteria for a corresponding force grab. If so, then at 3408

the object-grip module 3120 stores an indication that the virtual object being held by the user (which also represents the virtual object closest to the virtual hand) is being held via a force-type grab (e.g., force-pinch, force-grip, etc.). At 3410, the virtual object continues to be presented as being held by the virtual hand of the user. Furthermore, while 3406 describes determining whether the sensor data indicates that the capacitive grab-type now meets the criteria for a corresponding force grab, in some instances the process 3200(2) may also determine whether the sensor data meets the criteria for a different type of grab. For instance, if the current grab is a pinch-type grab, then the drop-initiation module 3130 or another component may determine whether the sensor data meets the criteria for a grip-type grab. Or, if the current grab is a grip-type grab, then the drop-initiation module 3130 or another component may determine whether the sensor data meets the criteria for a pinch-type grab.

[0225] Returning to the figure, if the drop-initiation module 3130 determines at the operation 3404 that the current grab type is not capacitive, or in response to the drop-initiation module 3130 determining that the sensor data does indicate that the grab meets the criteria of a corresponding force-grab type, then at an operation 3414 the drop-initiation module 3130 determines whether the sensor data indicates that the criteria for the current grab type are met. If so, then at 3416, the drop-initiation module 3130 determines where a difference in force between the current force reading and a previous force reading is greater than a force threshold. That is, the drop-initiation module 3130 determines whether the sensor data indicates that a significant drop of force has occurred at the controller. If so, then at 3418, the drop-initiation module 3130 initiates a release of the virtual object and, at the operation 3412, returns to the operation 3202. If, however, the difference in force is not greater than the force threshold, then the process proceeds back to the operation 3202.

[0226] If, however, the drop-initiation module 3130 determines at the operation 3414 that the sensor data indicates that the criteria for the current grab type are not met, then at an operation 3420 the drop-initiation module 3130 determines whether the current velocity of the handheld controller is greater than a velocity threshold (e.g., 1m/second, etc.). If so, then at the operation 3418, the drop-initiation module 3130 initiates the release (e.g., throw, drop, smash, etc.) of the object. If not, however, then at an operation 3422 the drop-initiation module 3130 determines whether the current grab type is a force-grab type. If not, then the grab type cannot be downgraded and, thus, the process 3200 proceeds to initiate release of the object at the operation 3418.

[0227] If, however, the drop-initiation module 3130 determines at the operation 3422 that the current grab type is a force-grab type, then at an operation 3424 the drop-initiation module 3130 determines whether a corresponding cap-based grab (e.g., pinch, grip, etc.) exists. If so, then at an operation 3426 an indication of the cap-type grab is stored. If the drop-initiation module 3130 determines at the operation 3424 that no corresponding cap-based grab exists, then the drop-initiation module 3130 initiates the release at the operation 3418 and thereafter returns to the operation 3202.

[0228] While FIG. 34 describes one example process 3200(2) for determining when to initiate a release of an object, it is to be appreciated that other processes may be. For instance, it is noted that a release may be initiated in direct response to determining that the criteria for a current grab type is no longer being met. That is, if the techniques determine, at operation 3414, that the criteria for the current grab type is no longer met, then the process may proceed directly to initiating the release of the held object at operation 3418. Thus, some or all of the operations 3416, 3420, 3422, and 3424 (potentially amongst other operations) may be optional in some circumstances.

[0229] Fig. 35 illustrates a process 3200(3) after a virtual object has been held and prior to when the virtual object has been released (e.g., immediately after initiation of the release). At an operation 3502, the drop-object module 3132 receives additional sensor data. At an operation 3504, the drop-object module 3132 determines whether this is the first frame after initiating the release of the object. If so, then at an operation 3506 the drop-object module 3132 determines whether the current velocity of the handheld controller 100 is greater than the previous velocity. If not, then at an operation 3508 the drop-object module 3132 proceeds to release the object by, for example, storing an indication that the object has been released and/or causing presentation of the release of the object. For example, the object may be illustrated as being dropped, thrown, smashed, or the like.

[0230] If, however, the drop-object module 3132 determines that the current velocity is greater than the previous velocity, then at an operation 3512 the drop-object module 3132 stores an indication of new peak value (e.g., the highest velocity since initiation of the release) and an indication of the floor value (e.g., the lowest velocity since initiation of the release). The drop-object module 3132 may also calculate a new ending value, comprising a value between the floor and peak values. In some instances, the ending value corresponds to an average of the peak and floor values. Thereafter, the process 3200(3) returns to the operation 3202.

[0231] In some instances, as noted above, the floor value may correspond to the lowest velocity of the handheld controller since the initiation of the release of the object. In other instances, meanwhile, the floor velocity may comprise some percentage thereof (e.g., 50%, 75%, etc.), a lowest velocity for a preceding amount of time (e.g., five frames) prior to the initiation of the release of the object, a constant value, or any other value. Further, while the ending value may be based on the value of the peak and floor velocities (e.g., an average of these two values, etc.), in other instances the ending value may be based on the peak velocity alone (e.g., half the peak velocity), the floor velocity (e.g., twice the floor velocity), or the like.

[0232] If, however, the drop-object module 3132 determines that the current frame is not the first frame since initiation of the release of the virtual object, then at an operation 3514 the drop-object module 3132 determines whether the predetermined amount of time since the initiation of the release has elapsed. In some instances, this time is measured in seconds, minutes, or the like, while in other instances it may be measured in a number of frames (e.g., five frames), or the like. If the drop-object module 3132 determines that the predetermined amount of time has elapsed without releasing the object, then the drop-object module 3132 may release the object at the operation 3508. If, however, the predetermined amount of time has not elapsed, then the drop-object module 3132 may proceed to determining, at an operation 3516, whether the current velocity is greater than the previous velocity (e.g., measured at the immediately subsequent frame). If so, then at an operation 3518 the drop-object module 3132 may determine whether the current velocity is greater than the current peak velocity. If so, then the process may proceed to the operation 3512, at which point the new peak value may be stored and the new ending value may be calculated. If, however, the current velocity is not greater than the peak value, then the process may return to the operation 3202.

[0233] If, however, the drop-object module 3132 determines at the operation 3516 that the current velocity is not greater than then previous velocity, then at an operation 3520 the drop-object module 3132 may determine whether the current velocity is less than the floor velocity. If so, then the drop-object module 3132 may release the object at the operation 3508. If not, then the drop-object module 3132 may determine whether the ending velocity is less than the current ending velocity and whether it has remained below the ending velocity for a predetermined amount of time. If so, then the drop-object module 3132 may proceed to release the object at the

operation 3508. If not, however, then the drop-object module 3132 may proceed to the operation 3510, representing a return to the operation 3202.

[0234] Fig. 36 illustrates different calculations 3602 that the drop-object module 3132 may make, with these calculations being used to determine how to present the release of a virtual object 3006, such as the speed, trajectory, landing location, and/or launch position of the virtual object.

[0235] Continuing the illustrated example from above, the virtual hand 3004 may be holding, and thereafter releasing, the virtual object 3006. This illustration further shows that the virtual object may be associated with a velocity (V) and a position (P) for each display frame (F). These velocities and positions may be based on sensor data received from one or more handheld controllers as described.

[0236] In the illustrated example of Fig. 36, the drop-initiation module 3130, the drop-object module 3132, and/or other software and/or hardware have determined, at a first frame (F_1), that an object is to be released. In some examples, the drop-object module 3132 may make this determination using the process 3500 or a similar process. Regardless of the trigger for the release of the virtual object 3006, upon making the determination to release the object the drop-object module may determine a position from which to release the virtual object 3006, a velocity at which to release the virtual object 3006, a trajectory along which the virtual object 3006 is to travel, a landing location of the virtual object 3006, and/or the like.

[0237] To make these determinations, the drop-object module 3132 may analyze sensor data from the handheld controller received for a predetermined time range prior to the time at which the drop-object module 3132 determined that the release is to occur (in this instance, F_1). For example, the drop-object module 3132 may analyze sensor data for a previous number of frames, such as fifteen frames prior to the release of the object. The drop-object module 3132 may then identify the frame associated with a highest velocity (i.e., a peak velocity) during this time period and, in some instances, may store an indication that the frame associated with this highest velocity corresponds to the frame at which the user intended to release the virtual object 3006. The drop-object module 3132 may then use information associated with this frame to determine how to present, on a display, the release of the virtual object (e.g., speed, direction, trajectory, etc.).

[0238] In one instances, the drop-object module 3132 may present the virtual hand 3004 releasing the virtual object 3006 at the peak velocity, while in other instances the drop-object module 3132 may compute an average velocity based on the average of the peak velocity, a velocity of the virtual object 3006 at a first number of frames preceding the peak-velocity frame, and a velocity of the virtual object 3006 at a second number of frames after the peak-velocity. For example, the drop-object module 3132 may calculate an average velocity of the peak velocity, the velocity of the virtual object 3006 at the frame immediately prior to the peak-velocity frame, and the velocity of the virtual object 3006 at the frame immediately after the peak-velocity frame.

[0239] Fig. 36, for instance, illustrates that the drop-object module 3132 has identified frame F_N as the frame within the predetermined amount of time preceding F_1 having the peak velocity, V_N . Therefore, in this example the drop-object module 3132 may identify a velocity, V_{N-1} , of the virtual object 3006 at a frame, F_{N-1} , immediately preceding F_N , and a velocity, V_{N+1} , of the virtual object 3006 at a frame, F_{N+1} , immediately after F_N . The drop-object module 3132 may then calculate an average velocity of these three frames and may present the virtual object 3006 being released from the virtual hand 3004 according to this average velocity.

[0240] In some instances, the drop-object module 3132 may present the virtual object 3006 being released according to the average velocity at a current position of the virtual object 3006. Fig. 37A, described below,

illustrates such an example. In other instances, meanwhile, the drop-object module 3132 may present the virtual object 3006 being released at the average velocity at a prior position. For example, the drop-object module 3132 may present the virtual hand 3004 releasing the virtual object 3006 at the average velocity, or at another velocity, from the position, P_1 , associated with the time at which the drop-object module 3132 determined that the object is to be released. Fig. 37B, also described below, illustrates such an example. In other instances, meanwhile, the drop-object module 3132 may present the virtual object 3006 being released at any frame prior to or after the frame associated with P_1 , such as from the position, P_N , of the virtual object 3006 at F_N , or from any other position (e.g., F_{N-1} , F_{N+1} , etc.).

[0241] In still other instances, the drop-object module 3132 may calculate a trajectory at which to present the virtual object travelling, and/or a landing location of the virtual object. For example, the drop-object module 3132 may calculate a trajectory of the virtual object 3006 if the virtual object 3006 had been released with the peak velocity and at the position of the peak velocity, P_N . The drop-object module 3132 may then present, on the display, the virtual object 3006 travelling along this calculated trajectory, either beginning at the point associated with the peak velocity (P_N) or beginning at the current position of the virtual object 3006 (e.g., P_1). In the latter instances, the drop-object module 3132 may effectively translate the calculated trajectory from the position, P_N , to the current position of the object, P_1 . Fig. 37C, described below, illustrates such an example.

[0242] In still other instances, the drop-object module 3132 may calculate a landing location (LL) of the virtual object 3006 had the virtual object 3006 been released at F_N . The drop-object module 3132 may then calculate a trajectory of the virtual object from its current position (e.g., P_1) to this landing location. The drop-object module 3132 may then present the virtual object 3006 traveling along this trajectory. In these instances, the virtual object 3006 will thus land at the location it would have landed if the object release had been identified exactly at F_N , while still avoiding the presentation of the object moving from its current position, such as P_1 , to a previous position, such as P_N , just prior to the object being released. Fig. 37D, described below, illustrates such an example.

[0243] Further, it is to be appreciated that the different types of corrective action described above may be introduced immediately or may be introduced after the determination to take the selected action. For instance, if drop-object module 3132 determines to release an object at the average velocity discussed above, the drop-object module 3132 may cause the object to speed up or slow down (as appropriate) from the current velocity of the object to the average velocity over time, rather than immediately jumping from the current velocity to the average velocity. Similarly, if the drop-object module 3132 determines to implement a different trajectory than the current trajectory of the object, the module 3132 may implement the different trajectory over time to avoid appearance of the object immediately jumping from one location to another on the display.

[0244] Fig. 37A illustrates an example where the drop-object module 3132 calculates a velocity at of the virtual object 3006 upon release, with the calculated velocity based at least in part on a peak velocity that occurs within a predetermined time range prior to determining that the user intends to release the virtual object 3006. For instance, the drop-object module 3132 may calculate this average velocity (V_{avg}) based on the average between the peak velocity, V_N , and the velocities associated with the frames on either side of the peak-velocity frame, V_{N-1} and V_{N+1} . It is to be appreciated that these velocities may represent both a magnitude of speed and a direction. Furthermore, regardless of which velocities are used to calculate V_{avg} , the drop-object module 3132 may present, on the display, the virtual object 3006 being released based at least in part on V_{avg} from the current

position of the virtual object, such as P_1 . As described above, this may comprise speeding up or slowing down (as appropriate) the current velocity towards V_{avg} , potentially until it reaches V_{avg} .

[0245] Fig. 37B illustrates an example where the drop-object module 3132 determines a position of the virtual object 3006 at a time at which the user intended to release the object 3006, as well as the drop-object module presenting the virtual object 3006 being released from that position. For example, and as described above, the drop-object module 3132 may store an indication that the intended release point of the virtual object occurred at the position, P_N , associated with the frame, F_N , having the peak velocity, V_N , within the predetermined amount of time from the current time (e.g., the fifteen frames prior to F_1). The drop-object module 3132 may then cause the display to present the release of the object from this position (P_N) at or otherwise based on a particular velocity, such as the peak velocity, the average velocity discussed above with reference to Fig. 37A, or any other velocity. As described above, this may comprise speeding up or slowing down (as appropriate) the current velocity towards the selected velocity, potentially until it reaches the selected velocity.

[0246] Fig. 37C illustrates an example where the drop-object module 3132 calculates a trajectory of the virtual object 3006 if the virtual object 3006 had been released at the time the user intended to release the virtual object 3006 and presenting the virtual object 3006 travelling along this trajectory, albeit beginning at a current position of the virtual object 3006 on the display. Stated otherwise, in this example the drop-object module 3132 may use the peak velocity, V_N , to determine the trajectory, T_N , that would have occurred if the virtual object 3006 had been presented as being released at that position, P_N . The drop-object module 3132 may then present the virtual object 3006 travelling a trajectory based on this trajectory, such as a trajectory that begins at the current position of the virtual object 3006 on the display, such as the position P_1 , but transitions toward the trajectory T_N . In this example, the displayed trajectory may be true to what the actual trajectory would have been, although the landing location may differ given that the starting position of the object release began at P_1 rather than P_N .

[0247] Fig. 37D illustrates an example where the drop-object module 3132 calculates a landing location of the virtual object 3006 based upon a trajectory of the virtual object 3006 if the object had been released at the time the user intended to release the virtual object. In this example, the drop-object module 3132 uses the peak velocity, V_N , and the position of the virtual object 3006 at the time of the peak velocity, P_N , to calculate the trajectory, T_N . This trajectory is then used, along with information regarding the environment of the virtual object 3006 to determine a landing location, LL_N . This landing location represents a position in the environment at which the virtual object 3006 would have landed if the virtual object 3006 had been presented as being released at F_N .

[0248] With this information, the drop-object module 3132 may then calculate a new trajectory between the current position of the virtual object (e.g., P_1) and the landing location, LL_N . While this solution alters the calculated trajectory of the virtual object, it allows display of the virtual object 3006 landing at the location it would have landed if released at F_N without presenting the virtual object 3006 to begin its release at a position associated with a previous frame (e.g., P_N).

[0249] Fig. 38 illustrates an example process 3800 of determining, at a first time, that a virtual hand is to release a virtual object and presenting, on a display, the virtual hand releasing the virtual object based at least in part on a peak velocity of the virtual object within a predetermined time range preceding the first time, or a position of the virtual object associated with the peak velocity. In some instances, the drop-object module 3132 may perform some or all of the process 3800.

[0250] At an operation 3802, the drop-object module 3132 may receive, while a display presents a virtual hand holding a virtual object sensor data from one or more sensors of a handheld controller. As described above, this sensor data may comprise force data, capacitance data, velocity data, and/or the like. At an operation 3804, the drop-object module 3132 may determine, at a first time and based at least in part on the sensor data, that the virtual hand is to release the virtual object. In some instances, the drop-object module 3132 may make this determination using the process 3200(3), while in other instances, it may use a different process.

[0251] At an operation 3806, the drop-object module 3132 may determine, at least partly in response to determining that the virtual hand is to release the virtual object, a peak velocity of the handheld controller within a predetermined time range prior to the first time. In some instances, the first time may be associated with a first frame, and this determining may comprise determining the peak velocity of the handheld controller within a predetermined number of frames immediately prior to the first frame, such as the previous fifteen frames.

[0252] At an operation 3808, the drop-object module 3132 may determine that the peak velocity within the predetermined time range occurred at a second time prior to the first time, such as at a particular frame prior to the first time. At an operation 3810, the drop-object module may present, on the display, the virtual hand releasing the virtual object based at least in part on at least one of a speed of the handheld controller at the second time, a direction of the handheld controller at the second time, or a position of the handheld controller at the second time, and/or based on a rotational velocity of the virtual object at the second.

[0253] In some instances, for example, the drop-object module 3132 may determine at least one of a velocity of the handheld controller at a third time prior to the second time or a velocity of the handheld controller at a fourth time after the second time and may calculate an average velocity (e.g., speed and direction) using these velocities. The drop-object module 3132 may then present the virtual hand releasing the virtual object according to or otherwise based on this average velocity. In other instances, the drop-object module 3132 may additionally, or alternatively, determine a position of the virtual object at the second time, and may present the virtual hand releasing the virtual object from or otherwise based on this second position.

[0254] In still other instances, the drop-object module 3132 may determine, from the peak velocity, a direction of the handheld controller at the second time. In these instances, the virtual object may be presented as being released according to or otherwise based on this direction of the handheld controller at the second time. In some instances, the drop-object module 3132 may additionally, or alternatively, calculate a trajectory and/or a landing location of the virtual object using the peak velocity and/or the position of the virtual object at the second time. The drop-object module may then present the virtual hand releasing the virtual object according to this trajectory or according to a trajectory that terminates at the landing location.

[0255] In still other instances, the drop-object module may present the virtual object being released based at least in part on a rotational velocity of the virtual object. To determine the rotational velocity, the drop-object module may initially determine a rate of rotational change of the handheld controller at a time corresponding to or proximate to the peak velocity of the handheld controller. In addition, the drop-object module or another module may determine a length between the handheld controller (e.g., a center-of-mass of the handheld controller) and a center-of-mass of the virtual object. This length, coupled with the rate of rotational change of the handheld controller, may be used to calculate the rotational velocity of the virtual object at or near the time of the peak velocity of the handheld controller. The drop-object module may then present the virtual object being released based at least in part on this rotational velocity. For instance, the amount of spin of object may be based on this

value. To provide an example, if a user is holding a virtual hammer, the drop-object module may determine the rate of rotational of the virtual hammer at the time of the release of the object and may use this information in determining how to present the release of the virtual hammer.

[0256] Fig. 39 illustrates an example process 3900 of determining, at a first time, that a virtual object is to be released and presenting, on a display, the virtual object being released based at least in part on a peak velocity of the virtual object within a predetermined time range preceding the first time, or a position of the virtual object associated with the peak velocity. In some instances, the drop-object module 3132 may perform some or all of the process 3900.

[0257] At an operation 3902, the drop-object module 3132 may determine, at a first time, that a virtual object presented on a display is to be released. At an operation 3904, the drop-object module 3132 may determine, at least partly in response to determining that the virtual object is to be released, a peak velocity of a handheld controller within a predetermined time range prior to the first time. At an operation 3906, the drop-object module 3132 may determine that the peak velocity within the predetermined time range occurred at a second time prior to the first time. At an operation 3908, the drop-object module 3132 may present, on the display, the virtual object being released based at least in part on at least one of a speed of the handheld controller at the second time, a direction of the handheld controller at the second time, or a position of the handheld controller at the second time, and/or based on a rotational velocity of the virtual object at the second.

[0258] For example, the drop-object module may present the virtual object being released based at least in part on a rotational velocity of the virtual object. To determine the rotational velocity, the drop-object module may initially determine a rate of rotational change of the handheld controller at a time corresponding to or proximate to the peak velocity of the handheld controller. In addition, the drop-object module or another module may determine a length between the handheld controller (e.g., a center-of-mass of the handheld controller) and a center-of-mass of the virtual object. This length, coupled with the rate of rotational change of the handheld controller, may be used to calculate the rotational velocity of the virtual object at or near the time of the peak velocity of the handheld controller. The drop-object module may then present the virtual object being released based at least in part on this rotational velocity. For instance, the amount of spin of object may be based on this value. To provide an example, if a user is holding a virtual hammer, the drop-object module may determine the rate of rotational of the virtual hammer at the time of the release of the object and may use this information in determining how to present the release of the virtual hammer.

[0259] Although the subject matter has been described in language specific to structural features, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features described. Rather, the specific features are disclosed as illustrative forms of implementing the claims.

[0260] The disclosure is described with reference to specific exemplary embodiments herein, but those skilled in the art will recognize that the disclosure is not limited to those. It is contemplated that various features and aspects of the disclosure may be used individually or jointly and possibly in a different environment or application. For example, features shown with regards to a right-hand controller may be implemented also in a left-hand controller, and vice versa. The specification and drawings are, accordingly, to be regarded as illustrative and exemplary rather than restrictive. For example, the word “preferably,” and the phrase “preferably but not necessarily,” are used synonymously herein to consistently include the meaning of “not necessarily” or optionally. “Comprising,” “including,” and “having,” are intended to be open-ended terms.

CLAIMS

What is claimed is:

1. A method comprising:

receiving first data from one or more sensors of a handheld controller, the first data indicating at least one of a force or proximity of at least a portion of a hand of a user holding the handheld controller at a first time;

storing, based at least in part on the first data, a first indication that a virtual object rendered on a display has been picked up by the user;

presenting, on the display and based at least in part on the first indication, a virtual hand of the user holding the virtual object;

receiving second data from the one or more sensors, the second data indicating at least one of a force or proximity of at least a portion of the hand at a second time;

storing, based at least in part on the second data, a second indication that the virtual hand is to release the virtual object within a predetermined amount of time from the second time;

receiving third data from the one or more sensors, the third data indicating a velocity of the handheld controller at a third time; and

presenting, on the display, the virtual hand releasing the virtual object prior to or upon expiration of the predetermined amount of time.

2. The method as recited in claim 1, wherein the first data indicates a force of at least a portion of the hand at the first time, and further comprising:

determining that the force at the first time is greater than a force threshold;

and wherein the storing the first indication comprises storing the first indication that the virtual object rendered on the display has been picked up by the user based at least in part on determining that the force at the first time is greater than the force threshold.

3. The method as recited in claim 1, wherein the first data comprises a first capacitance value measured by the one or more sensors, and further comprising:

determining that at least one of the first capacitance value or a second capacitance value that is based at least in part on the first capacitance value is greater than a capacitance threshold;

and wherein the storing the first indication comprises storing the first indication that the virtual object rendered on the display has been picked up by the user based at least in part on determining that the at least one of the first capacitance value or the second capacitance value is greater than the capacitance threshold.

4. The method as recited in claim 1, wherein:

the first data indicates a force of at least a portion of the hand at the first time;

the second data indicates a force of at least a portion of the hand at the second time; and

the method further comprises:

determining a difference between the force at the first time and the force at the second time; and

determining that the difference is greater than a difference threshold;

and wherein storing the second indication comprises storing the second indication that the hand is to release the virtual object based at least in part on determining that the difference is greater than the difference threshold.

5. The method as recited in claim 1, wherein the second data further indicates a velocity of the handheld controller at the second time, and further comprising:

determining that the velocity of the handheld controller at the second time is greater than a velocity threshold;

and wherein storing the second indication comprises storing the second indication that the hand is to release the virtual object based at least in part on determining that the velocity of the handheld controller at the second time is greater than the velocity threshold.

6. The method as recited in claim 1, wherein the second data comprises a first capacitance value measured by the one or more sensors, and further comprising:

determining that at least one of the first capacitance value or a second capacitance value that is based at least in part on the first capacitance value is not greater than a capacitance threshold;

and wherein storing the second indication comprises storing the second indication that the hand is to release the virtual object based at least in part on determining that the first capacitance value is not greater than the capacitance threshold.

7. The method as recited in claim 1, wherein the second data comprises a velocity of the handheld controller at the second time, and further comprising:

determining that the velocity of the handheld controller at the third time is not greater than the velocity of the handheld controller at the second time;

and wherein the presenting the virtual hand releasing the object comprises presenting, on the display, the virtual hand releasing the object based at least in part on determining that the velocity of the handheld controller at the third time is not greater than the velocity of the handheld controller at the second time.

8. The method as recited in claim 1, further comprising:

determining that the predetermined amount of time has expired without presenting the virtual hand releasing the virtual object;

and wherein the presenting the virtual hand releasing the object comprises presenting, on the display, the virtual hand releasing the object based at least in part on determining that the predetermined amount of time has expired without presenting the virtual hand releasing the virtual object.

9. The method as recited in claim 1, wherein the second data comprises a velocity of the handheld controller at the second time, and further comprising:

determining that the velocity of the handheld controller at the third time is greater than the velocity of the handheld controller at the second time;

storing an indication that the velocity of the handheld controller at the third time corresponds to a peak velocity;

storing an indication that the velocity of the handheld controller at the second time corresponds to a floor velocity;

calculating, based at least in part on the peak velocity and the floor velocity, an ending velocity; and
storing an indication of the ending velocity.

10. The method as recited in claim 9, further comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a fourth time; and

determining that the velocity of the handheld controller at the fourth time is less than the floor velocity;

and wherein the presenting the virtual hand releasing the virtual object comprises presenting the virtual hand releasing the virtual object based at least in part on determining that the velocity of the handheld controller at the fourth time is less than the floor velocity.

11. The method as recited in claim 9, further comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a fourth time;

determining that the velocity of the handheld controller at the fourth time is not less than the floor velocity;

determining that the velocity of the handheld controller at the fourth time is less than the ending velocity;

and

determining that a velocity of the handheld controller has remained less than the ending velocity for greater than a threshold amount of time;

and wherein the presenting the virtual hand releasing the virtual object comprises presenting the virtual hand releasing the virtual object based at least in part on determining that the velocity of the handheld controller has remained less than the ending velocity for greater than the threshold amount of time.

12. A system comprising:

one or more processors; and

one or more computer-readable media storing computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving first data from one or more sensors of a handheld controller, the first data indicating a force or proximity of at least a portion of a hand of a user holding the handheld controller;

storing an indication that a virtual object is to be held based at least in part on the first data;

receiving second data from the one or more sensors, the second data comprising a velocity of the handheld controller; and

storing an indication that the virtual object is to be released based at least in part on the second data.

13. The system as recited in claim 12, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

determining that the velocity of the handheld controller is less than a previously measured velocity of the handheld controller;

and wherein the storing the indication that the virtual object is to be released comprises storing the indication that the virtual object is to be released based at least in part on determining that the velocity of the handheld controller is less than the previously measured velocity of the handheld controller.

14. The system as recited in claim 12, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

determining that a predetermined amount of time has elapsed since storing the indication that the virtual object is to be released; and

causing a display to present release of the virtual object based at least in part on determining that the predetermined amount of time has elapsed.

15. The system as recited in claim 12, wherein the velocity of the handheld controller comprises a velocity of the handheld controller at a first time, and the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving third data from the one or more sensors, the third data comprising a velocity of the handheld controller at a second time that is after the first time;

determining that the velocity of the handheld controller at the second time is not greater than the velocity of the handheld controller at the first time; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller at the second time is not greater than the velocity of the handheld controller at the first time.

16. The system as recited in claim 12, wherein the velocity of the handheld controller comprises a velocity of the handheld controller at a first time, and the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving third data from the one or more sensors, the third data comprising a velocity of the handheld controller at a second time that is after the first time;

determining that the velocity of the handheld controller at the second time is greater than the velocity of the handheld controller at the first time;

storing an indication that the velocity of the handheld controller at the second time corresponds to a peak velocity;

storing an indication that the velocity of the handheld controller at the first time corresponds to a floor velocity;

calculating an ending velocity based at least in part on at least one the peak velocity and the floor velocity; and

storing an indication of the ending velocity.

17. The system as recited in claim 16, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a third time;

determining that the velocity of the handheld controller at the third time is less than the floor velocity; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller at the third time is less than the floor velocity.

18. The system as recited in claim 16, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a third time;

determining that the velocity of the handheld controller at the third time is not less than the floor velocity;

determining that the velocity of the handheld controller at the third time is less than the ending velocity;

determining that a velocity of the handheld controller has remained less than the ending velocity for greater than a threshold amount of time; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller has remained less than the ending velocity for greater than the threshold amount of time.

19. A handheld controller comprising:

a controller body;

one or more sensors coupled to the controller body;

one or more processors; and

one or more computer-readable media storing computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving first data from the one or more sensors, the first data indicating a force or proximity of at least a portion of a hand of a user holding the handheld controller;

storing an indication that a virtual object is to be held based at least in part on the first data;

receiving second data from the one or more sensors, the second data comprising a velocity of the handheld controller; and

storing an indication that the virtual object is to be released based at least in part on the second data.

20. The handheld controller as recited in claim 19, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

determining that the velocity of the handheld controller is less than a previously measured velocity of the handheld controller;

and wherein the storing the indication that the virtual object is to be released comprises storing the indication that the virtual object is to be release based at least in part on determining that the velocity of the handheld controller is less than the previously measured velocity of the handheld controller.

21. The handheld controller as recited in claim 19, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

determining that a predetermined amount of time has elapsed since storing the indication that the virtual object is to be released; and

causing a display to present release of the virtual object based at least in part on determining that the predetermined amount of time has elapsed.

22. The handheld controller as recited in claim 19, wherein the velocity of the handheld controller comprises a velocity of the handheld controller at a first time, and the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving third data from the one or more sensors, the third data comprising a velocity of the handheld controller at a second time that is after the first time;

determining that the velocity of the handheld controller at the second time is not greater than the velocity of the handheld controller at the first time; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller at the second time is not greater than the velocity of the handheld controller at the first time.

23. The handheld controller as recited in claim 19, wherein the velocity of the handheld controller comprises a velocity of the handheld controller at a first time, and the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving third data from the one or more sensors, the third data comprising a velocity of the handheld controller at a second time that is after the first time;

determining that the velocity of the handheld controller at the second time is greater than the velocity of the handheld controller at the first time;

storing an indication that the velocity of the handheld controller at the second time corresponds to a peak velocity;

storing an indication that the velocity of the handheld controller at the first time corresponds to a floor velocity;

calculating an ending velocity based at least in part on the peak velocity and the floor velocity; and

storing an indication of the ending velocity.

24. The handheld controller as recited in claim 23, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a third time;

determining that the velocity of the handheld controller at the third time is less than the floor velocity; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller at the third time is less than the floor velocity.

25. The handheld controller as recited in claim 23, wherein the one or more computer-readable media further store computer-executable instructions that, when executed, cause the one or more processors to perform acts comprising:

receiving fourth data from the one or more sensors, the fourth data indicating a velocity of the handheld controller at a third time;

determining that the velocity of the handheld controller at the third time is not greater than the velocity of the handheld controller at the second time;

determining that the velocity of the handheld controller at the third time is not less than the floor velocity;

determining that the velocity of the handheld controller at the third time is less than the ending velocity;

determining that a velocity of the handheld controller has remained less than the ending velocity for greater than a threshold amount of time; and

causing a display to present release of the virtual object based at least in part on determining that the velocity of the handheld controller has remained less than the ending velocity for greater than the threshold amount of time.

26. The handheld controller as recited in claim 19, wherein the one or more sensors comprise:

an accelerometer configured to determine the velocity of the handheld controller;

a force sensing resistor (FSR) configured to generate data indicating force of at least a portion of the hand of the user on the handheld controller; and

a proximity sensor configured to generate data indicating proximity of at least a portion of the hand of the user on the handheld controller.

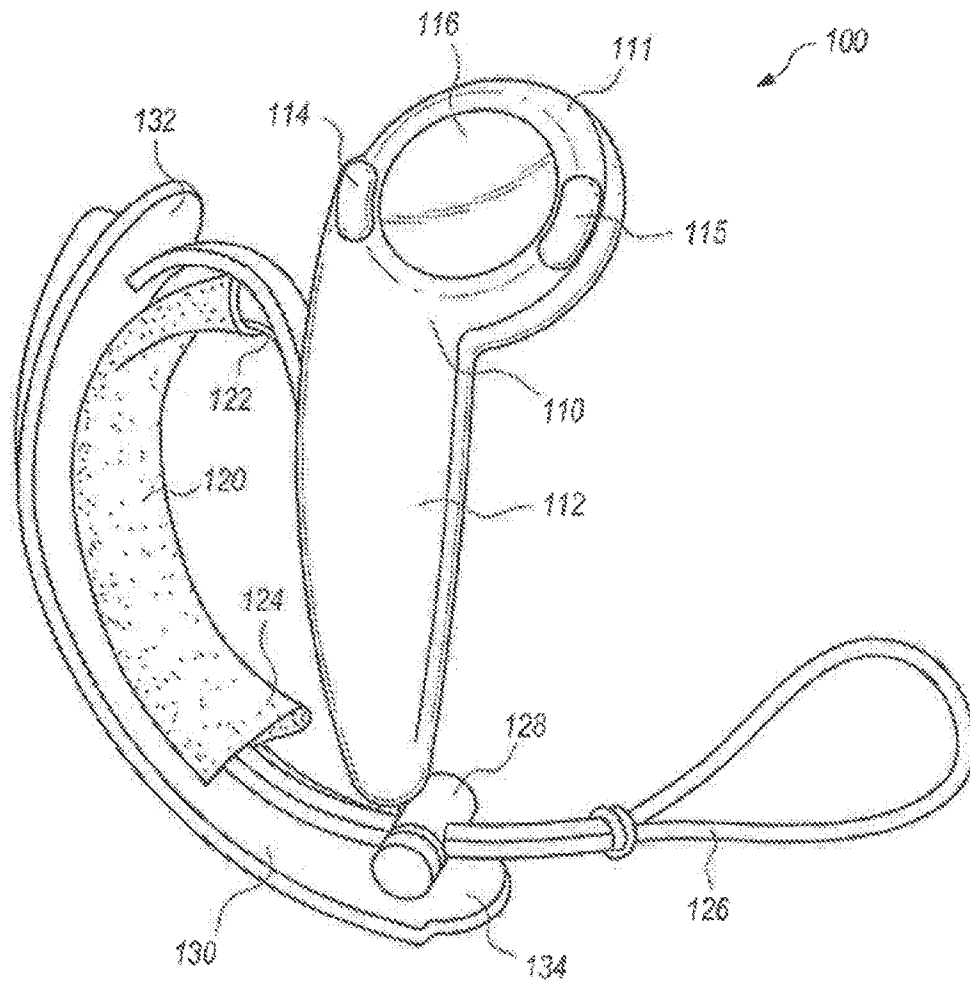


FIG. 1

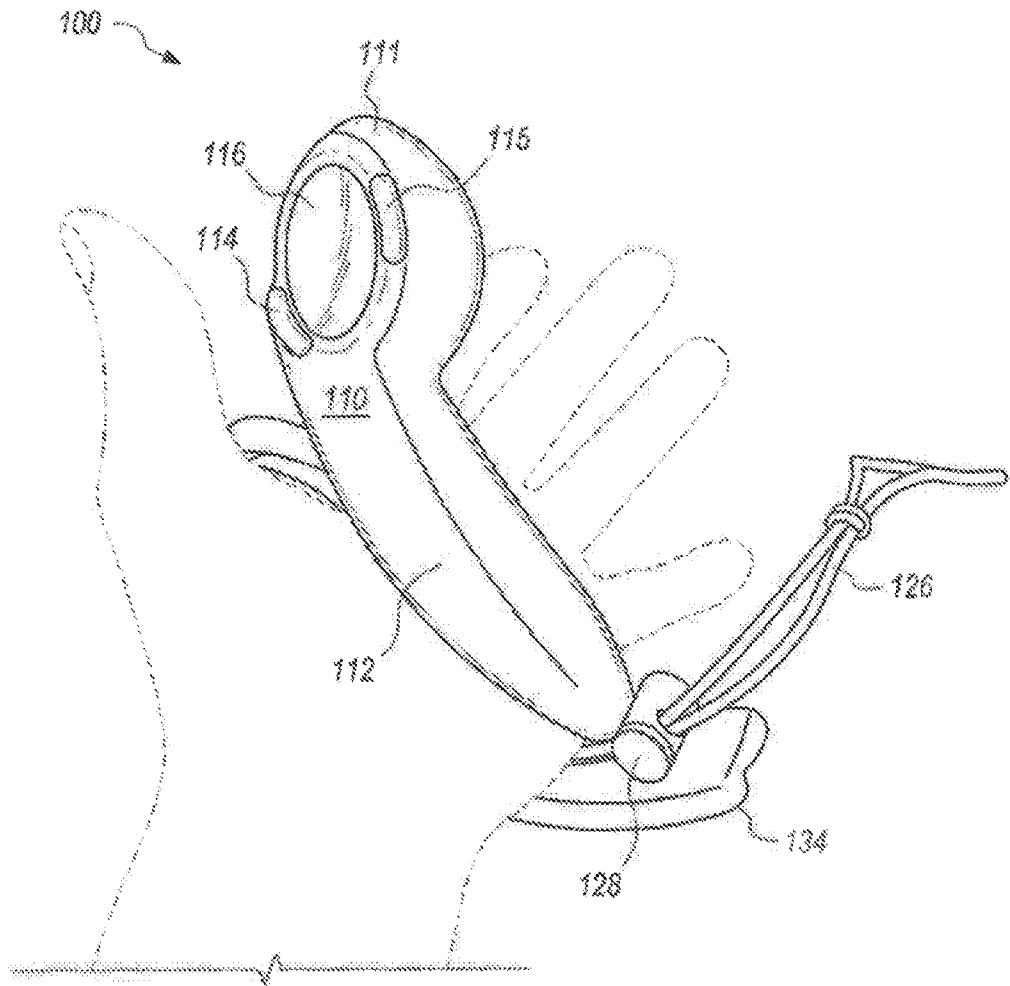


FIG. 2

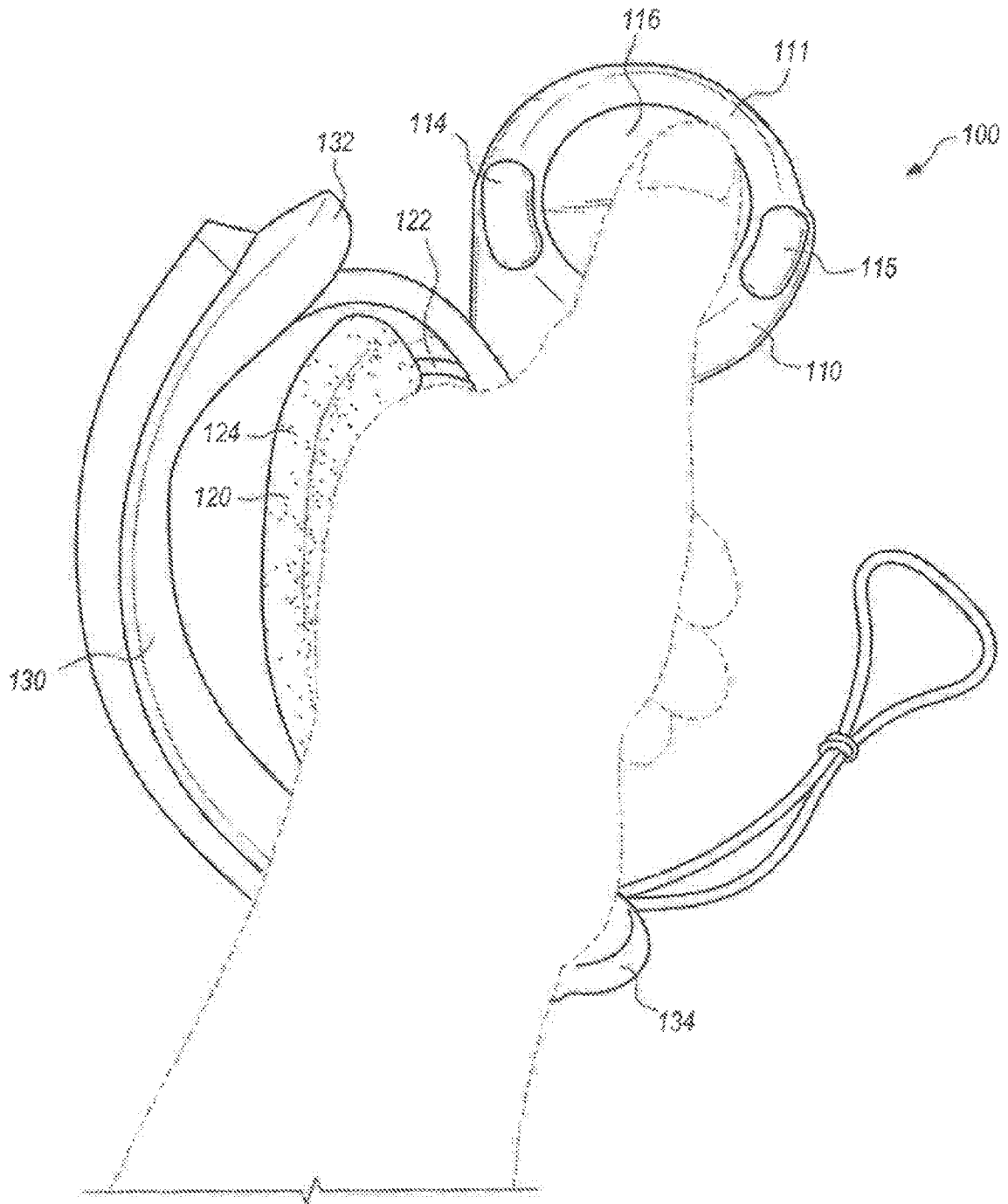


FIG. 3

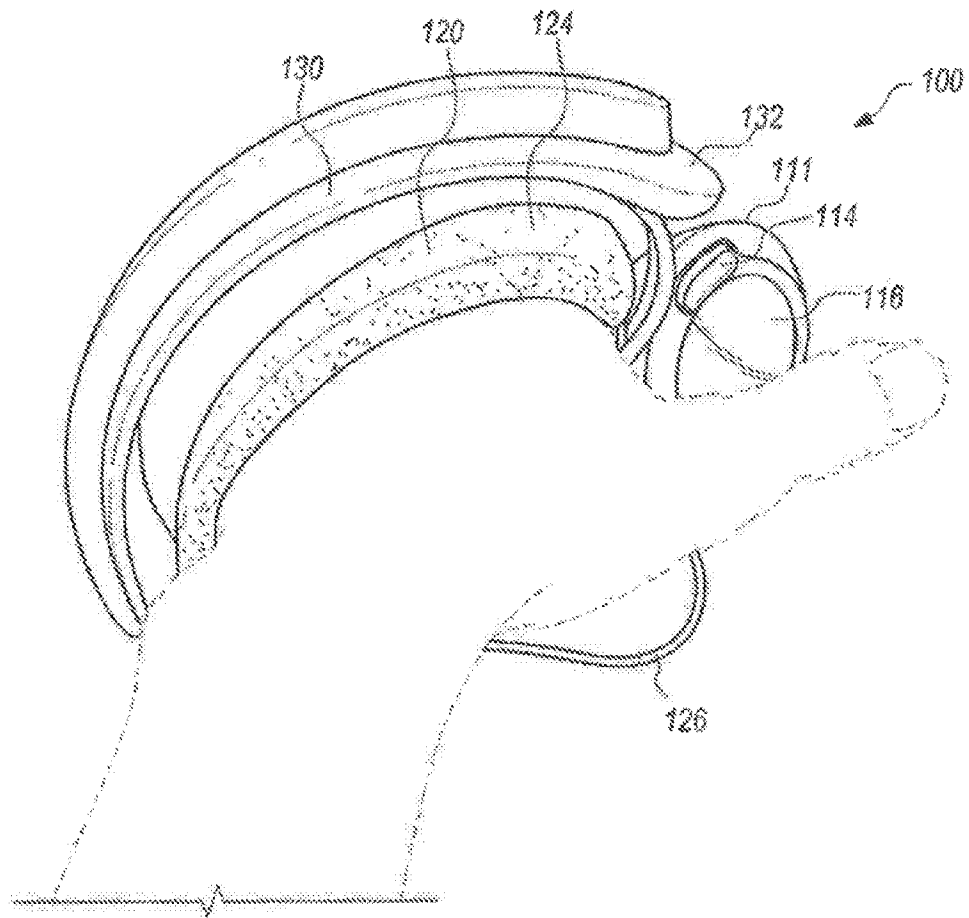


FIG. 4

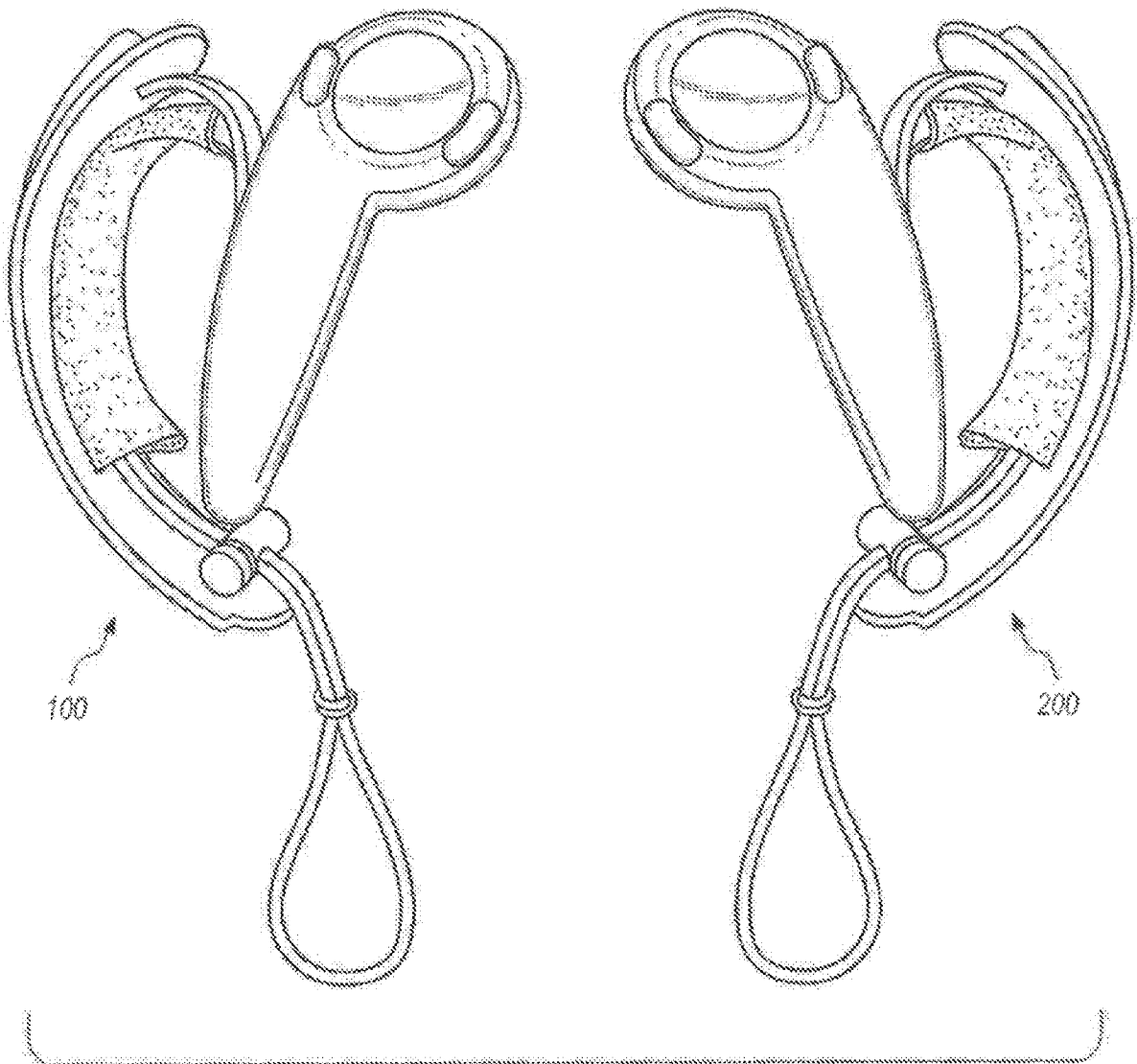


FIG. 5

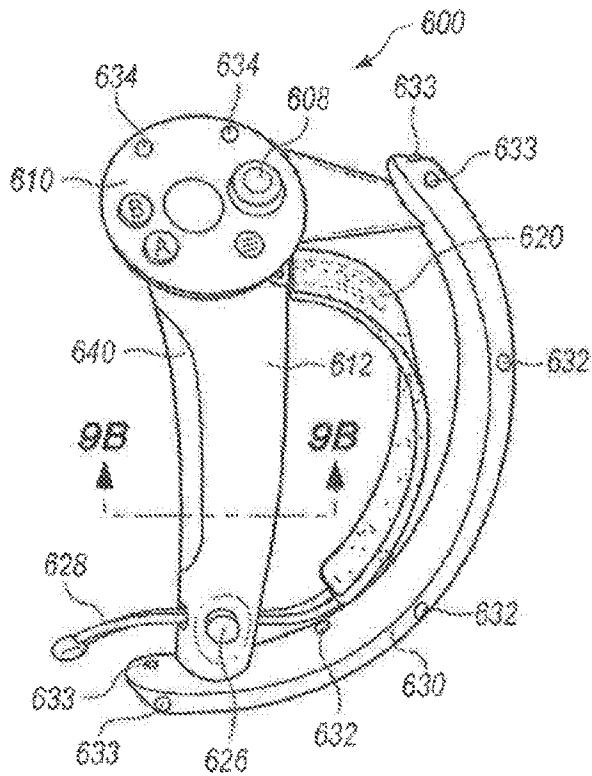


FIG. 6A

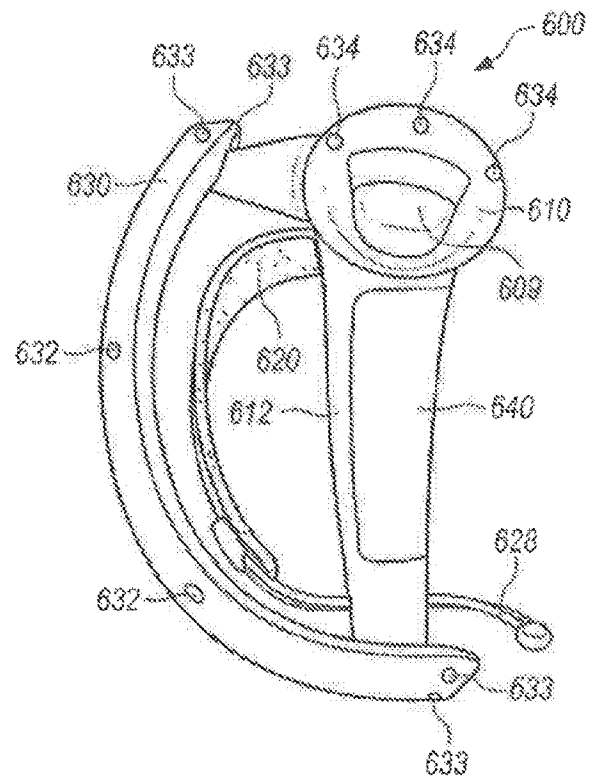


FIG. 6B

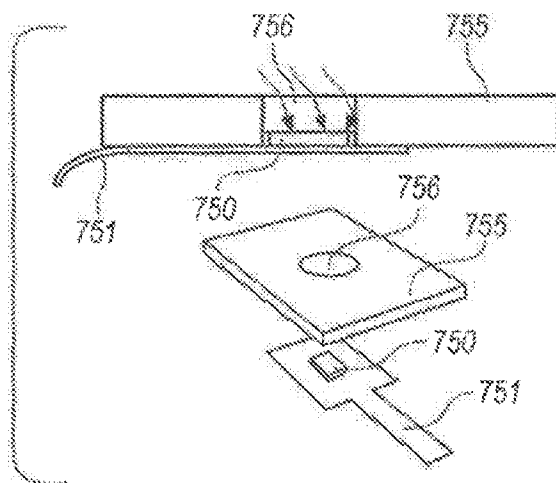


FIG. 7A

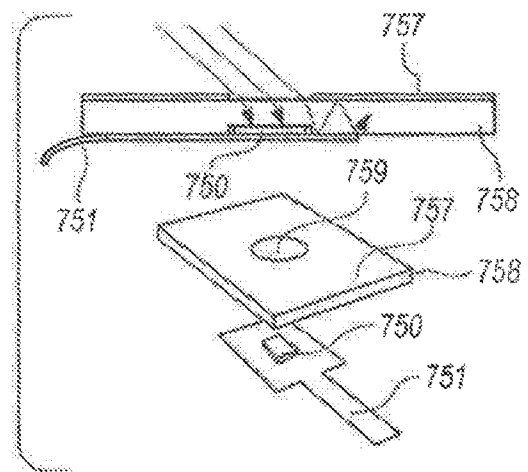


FIG. 7B

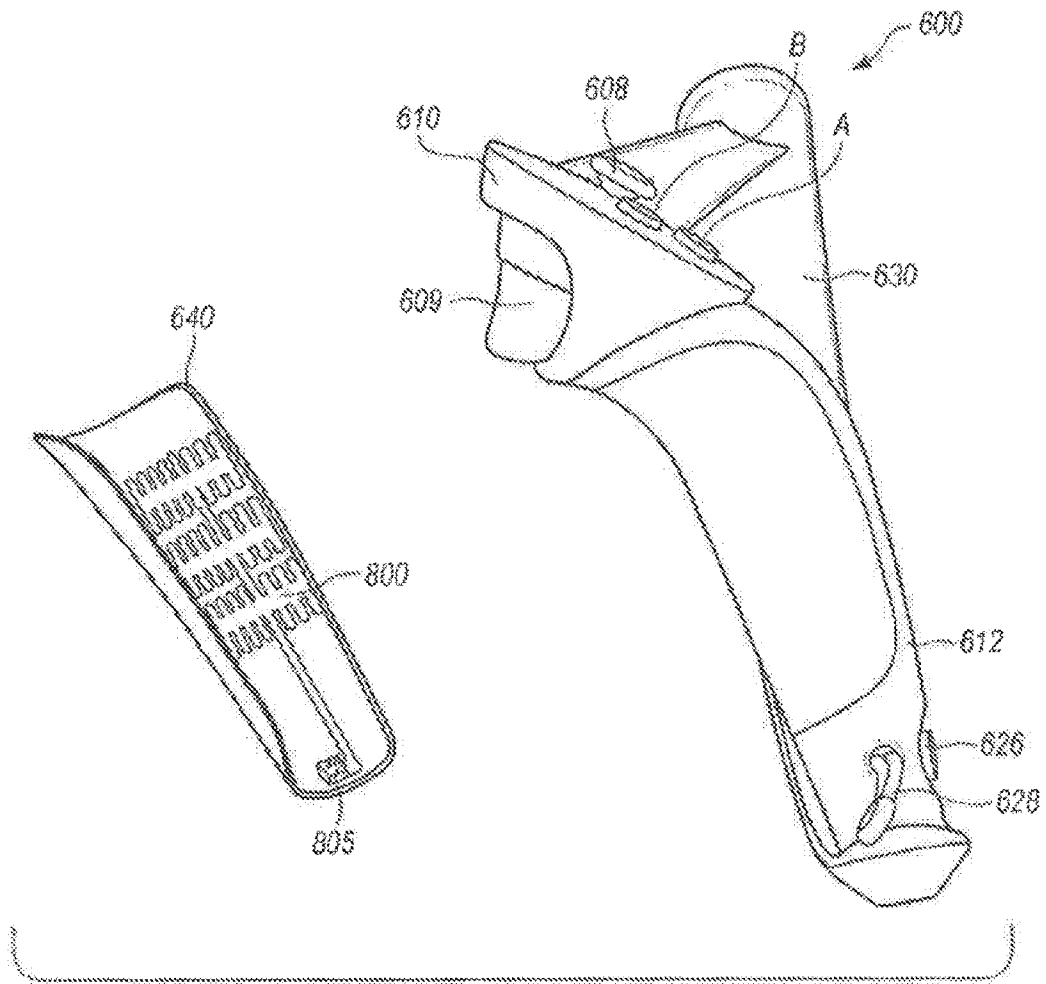


FIG. 8

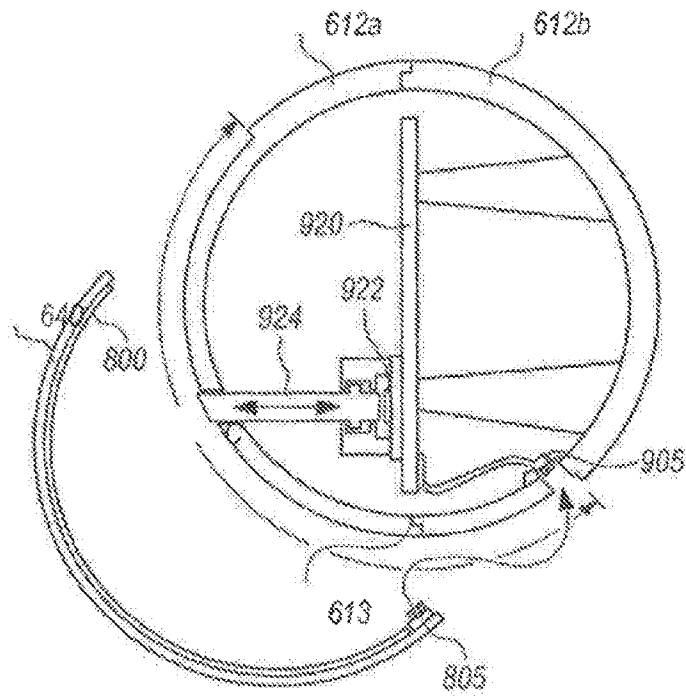


FIG. 9A

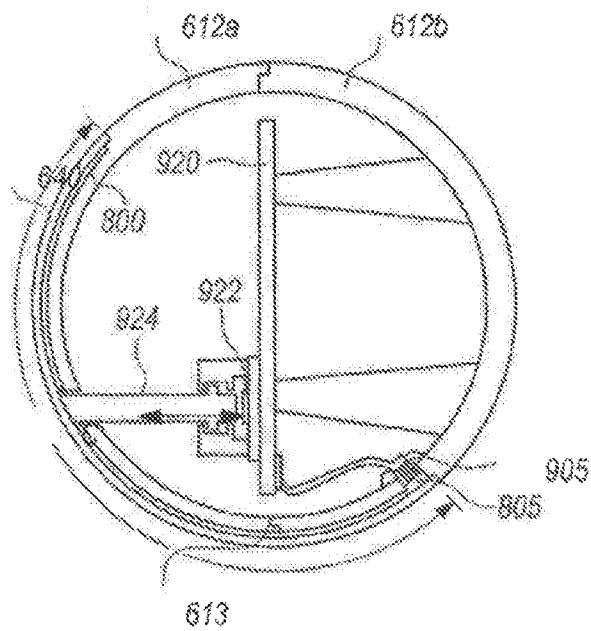


FIG. 9B

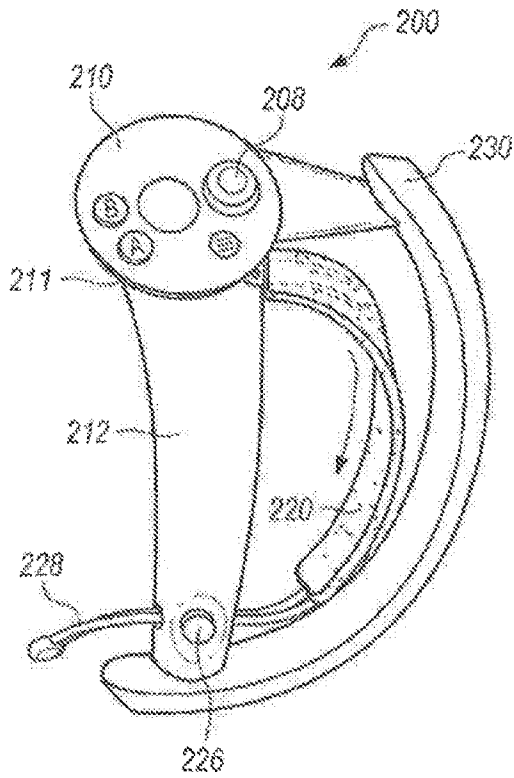


FIG. 10A

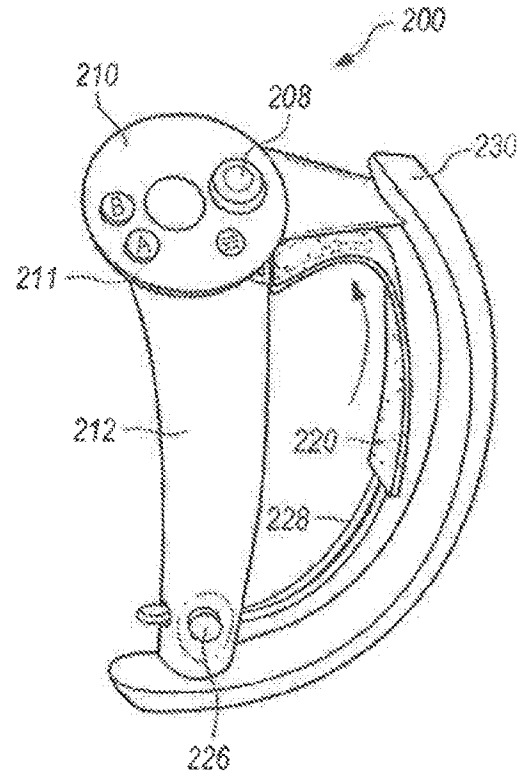


FIG. 10B

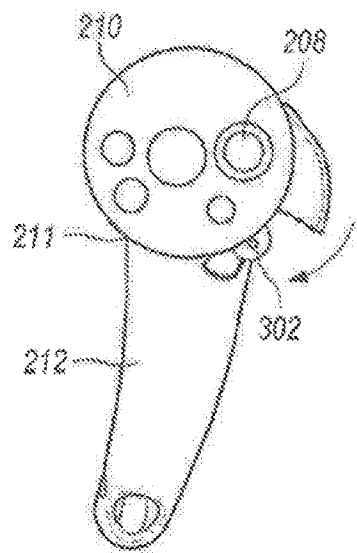


FIG. 11A

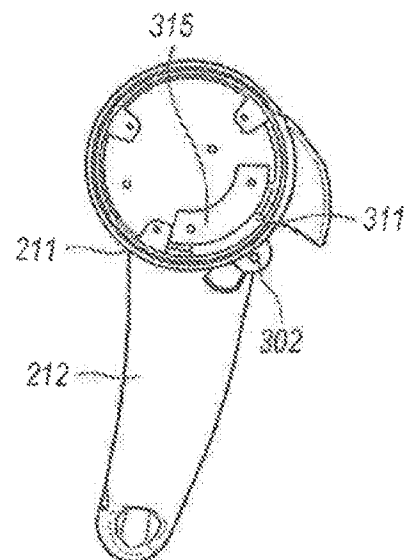
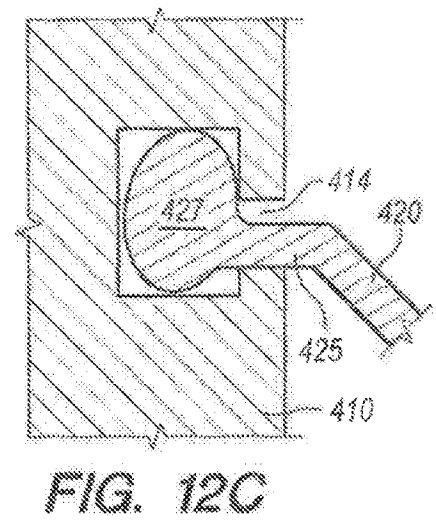
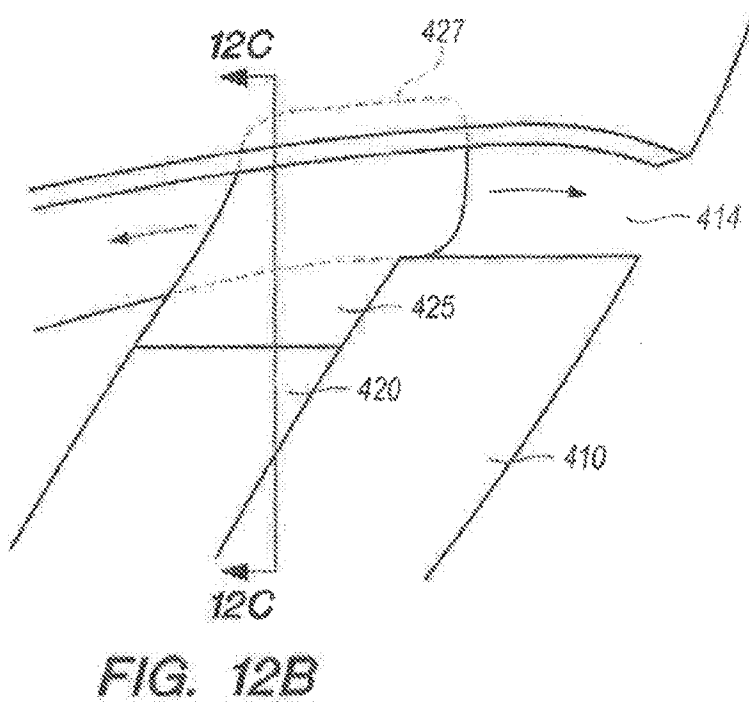
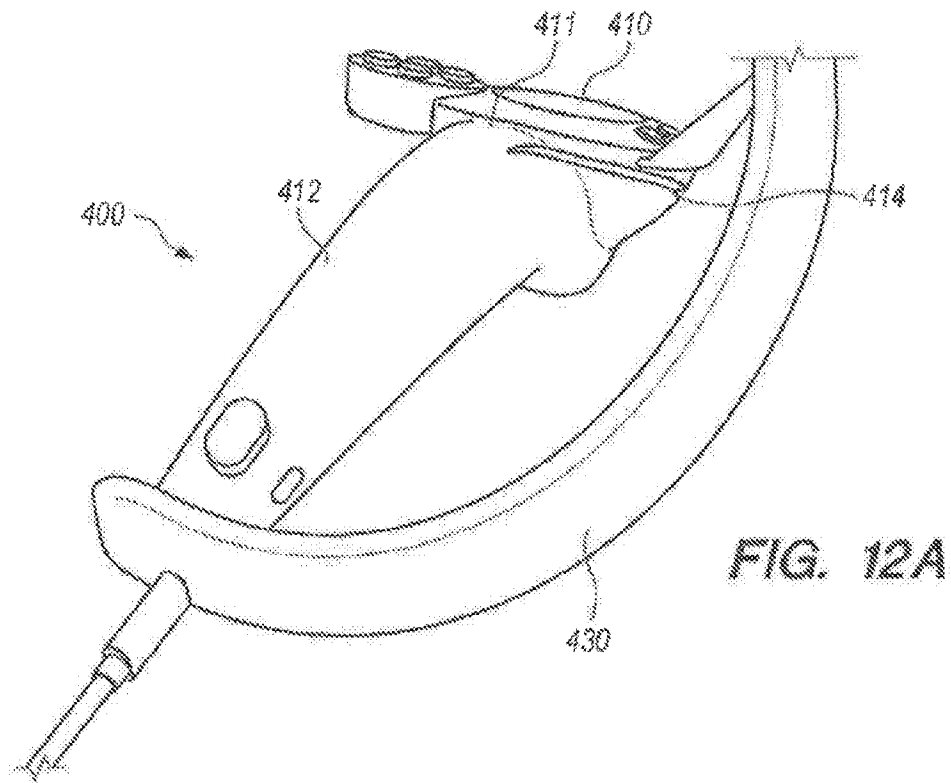


FIG. 11B



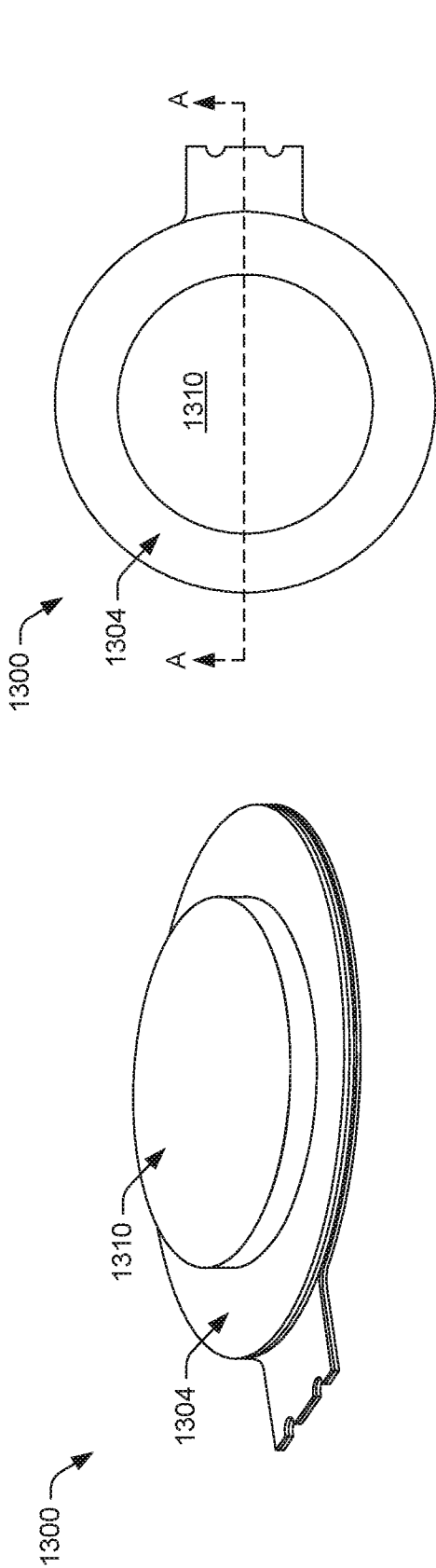
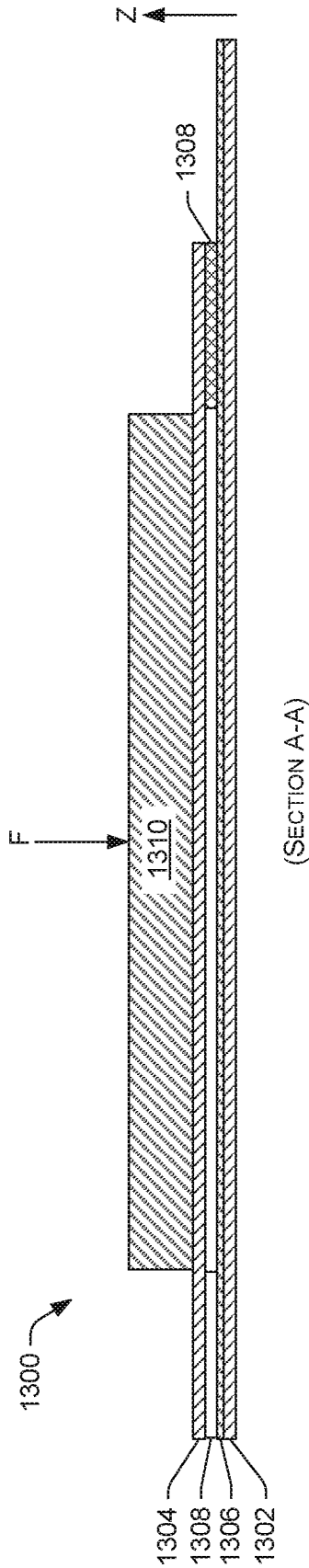


FIG. 13B

FIG. 13A



(SECTION A-A)

FIG. 13C

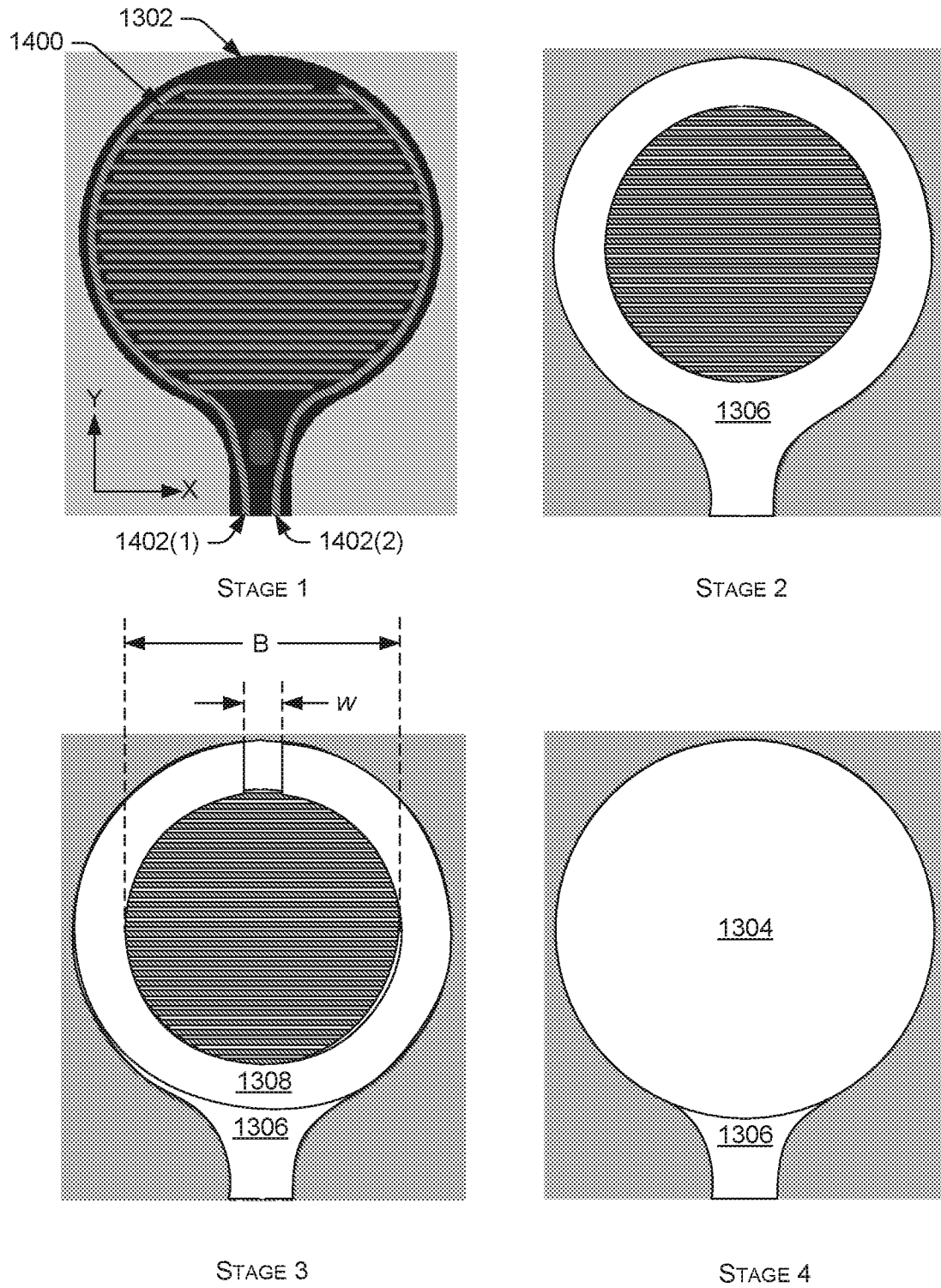


FIG. 14

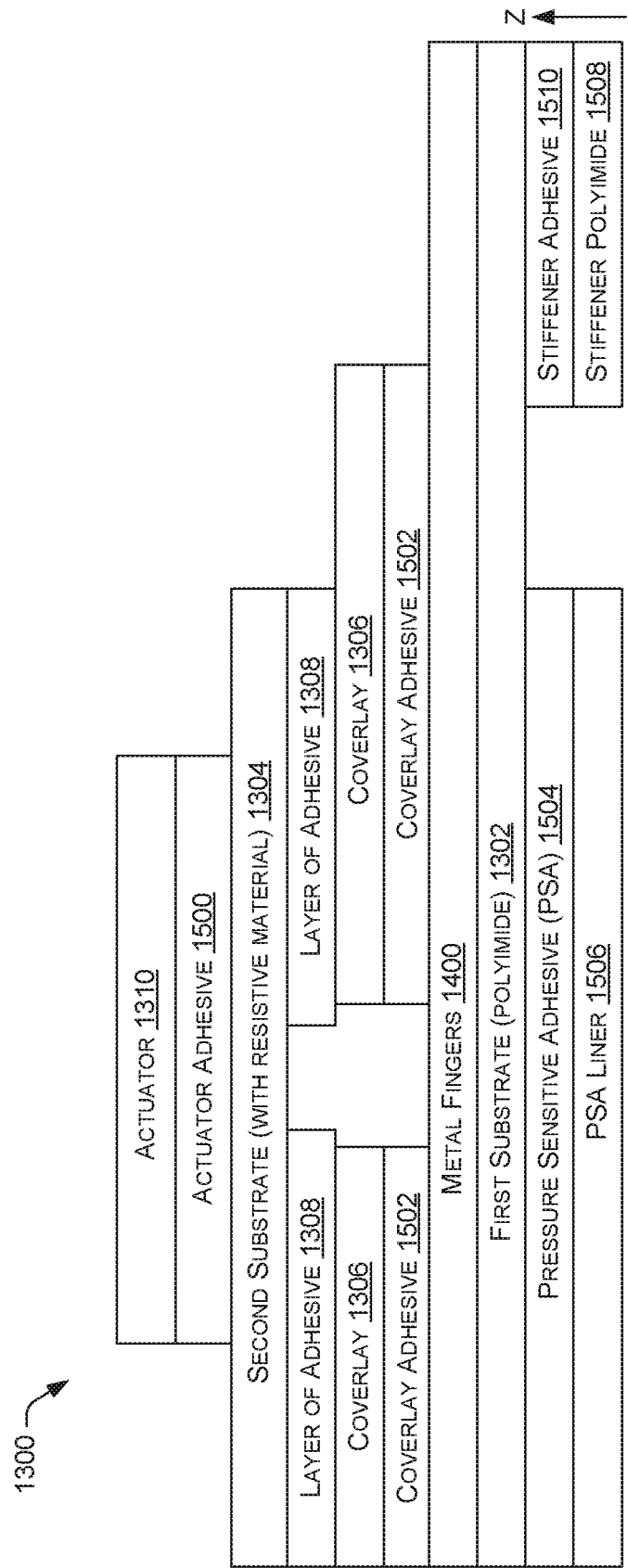


FIG. 15

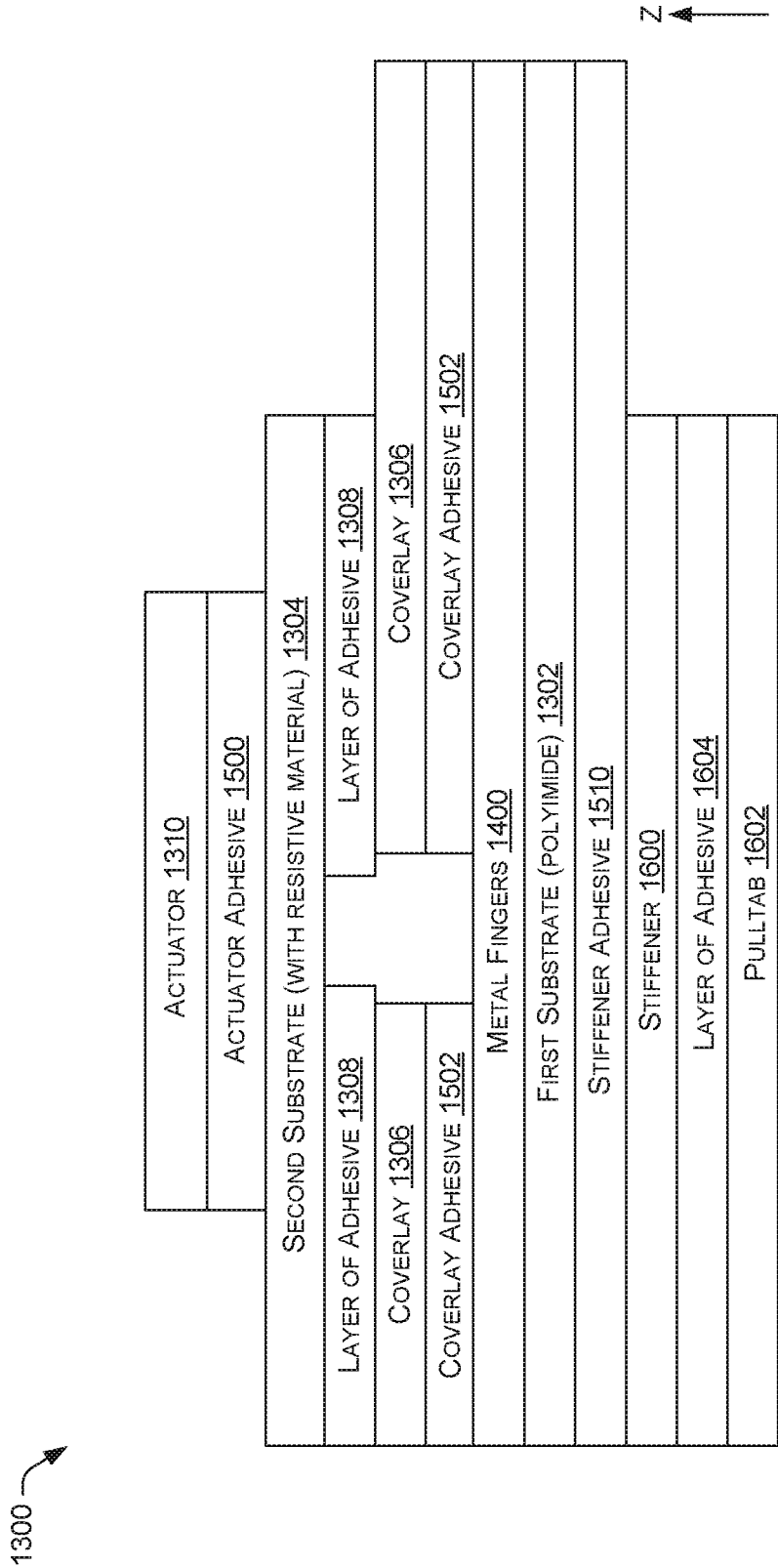


FIG. 16

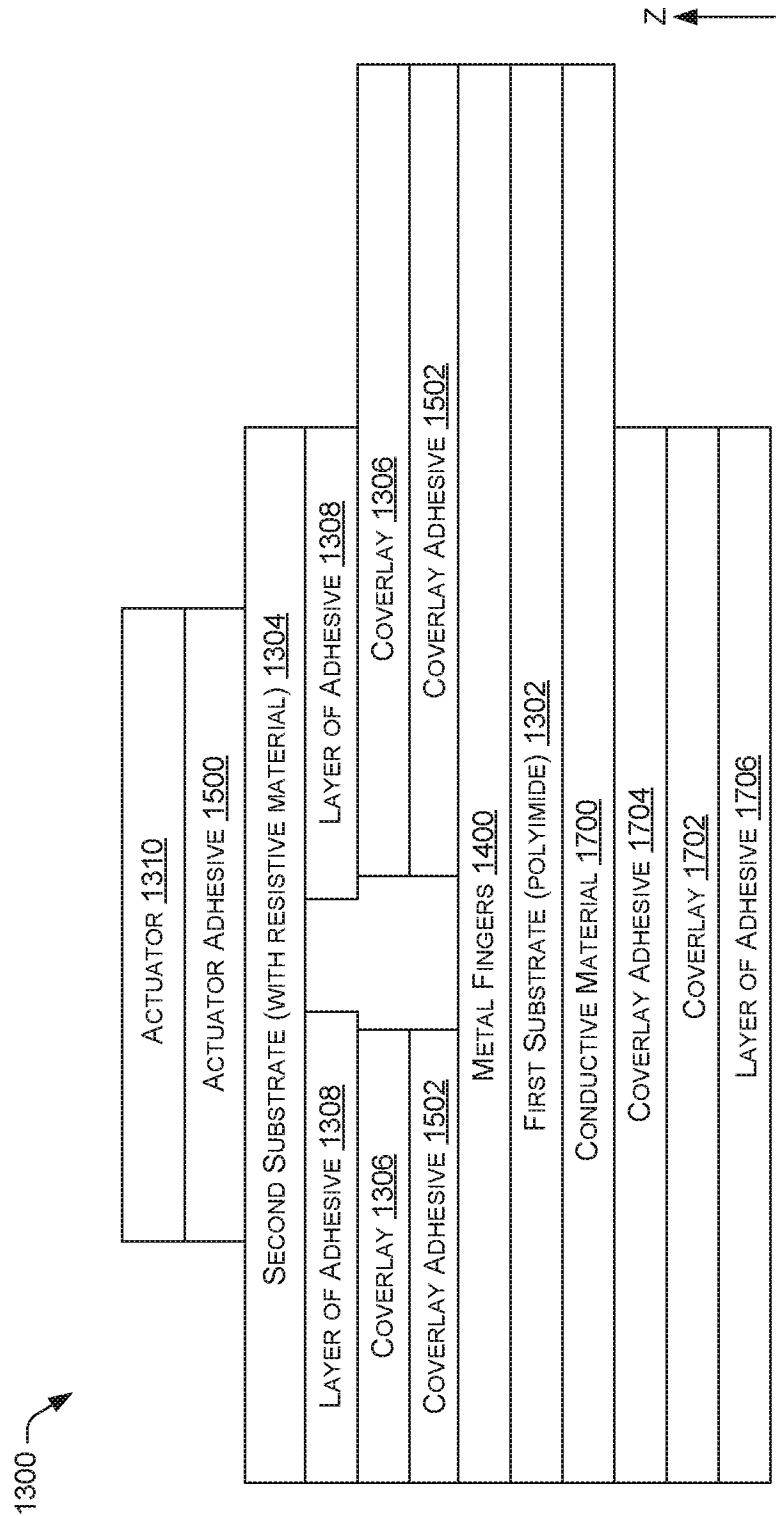
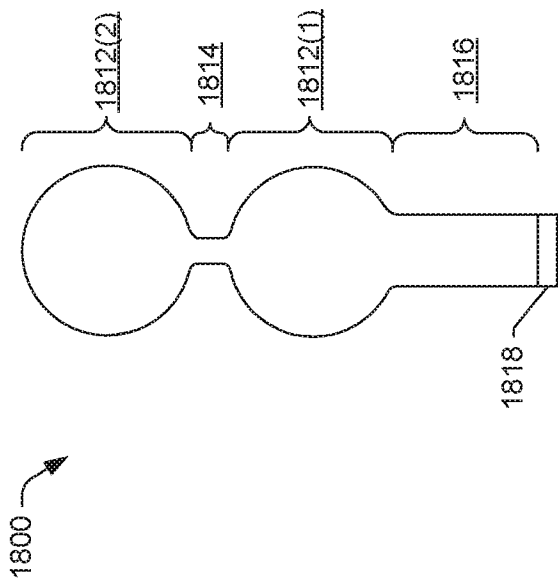
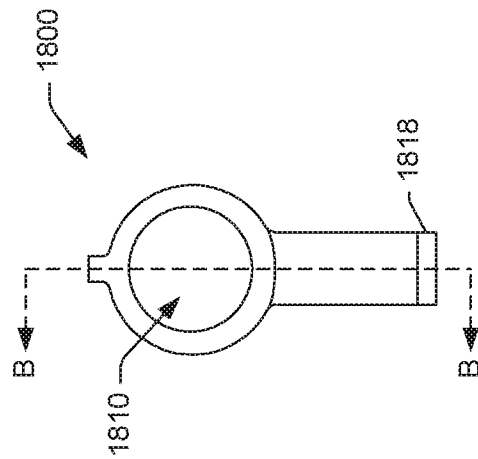


FIG. 17



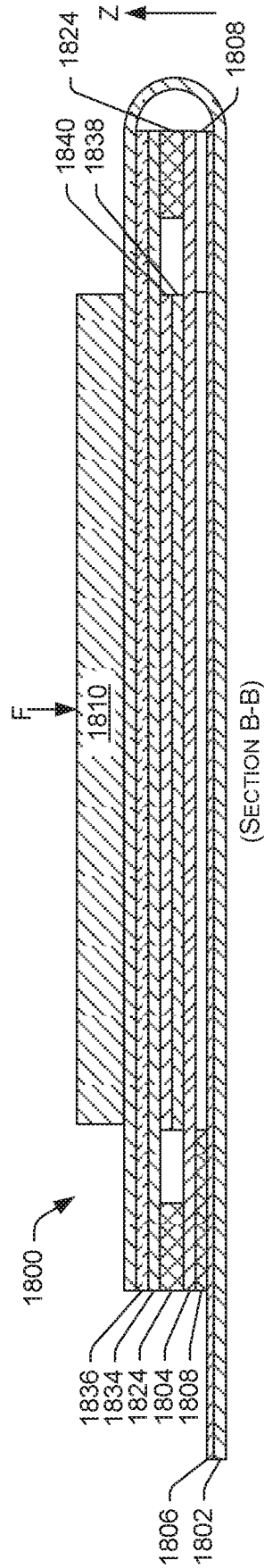
BEFORE FOLD

FIG. 18A



AFTER FOLD

FIG. 18B



(SECTION B-B)

FIG. 18C

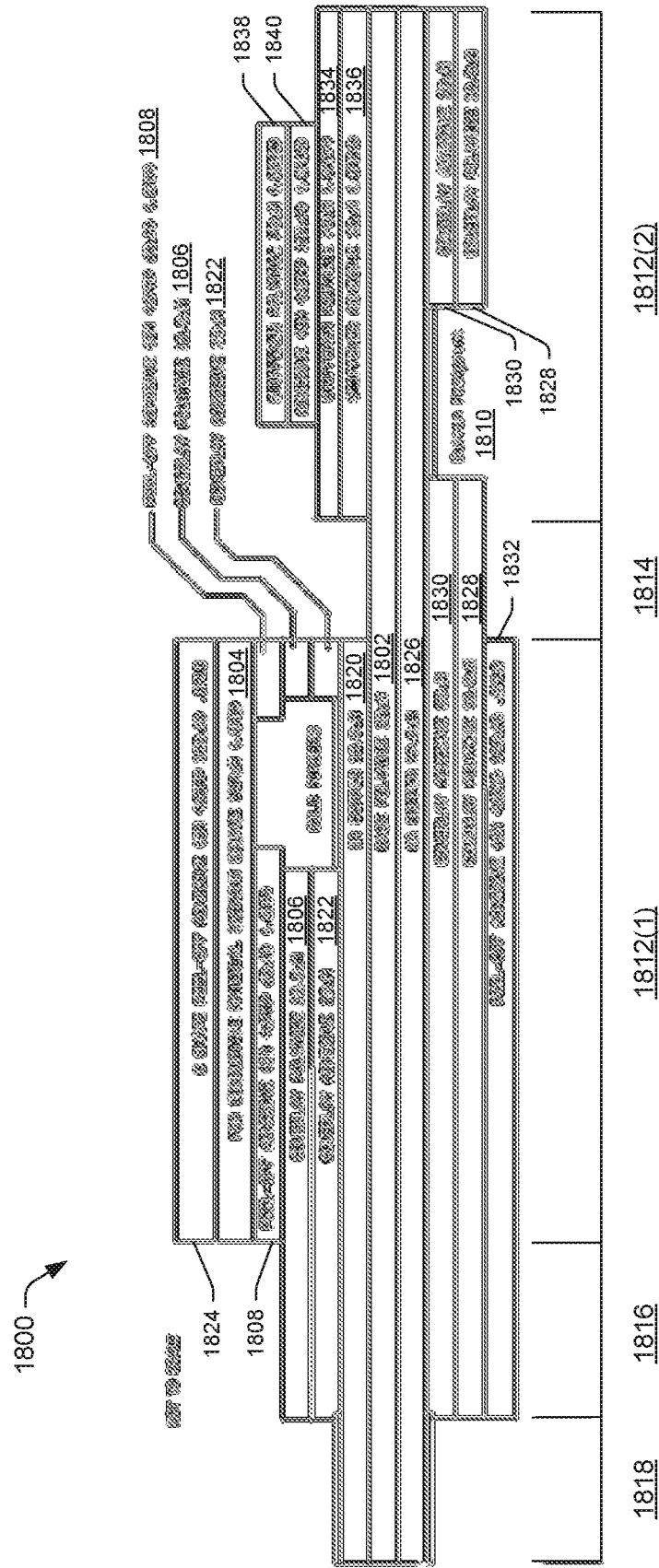
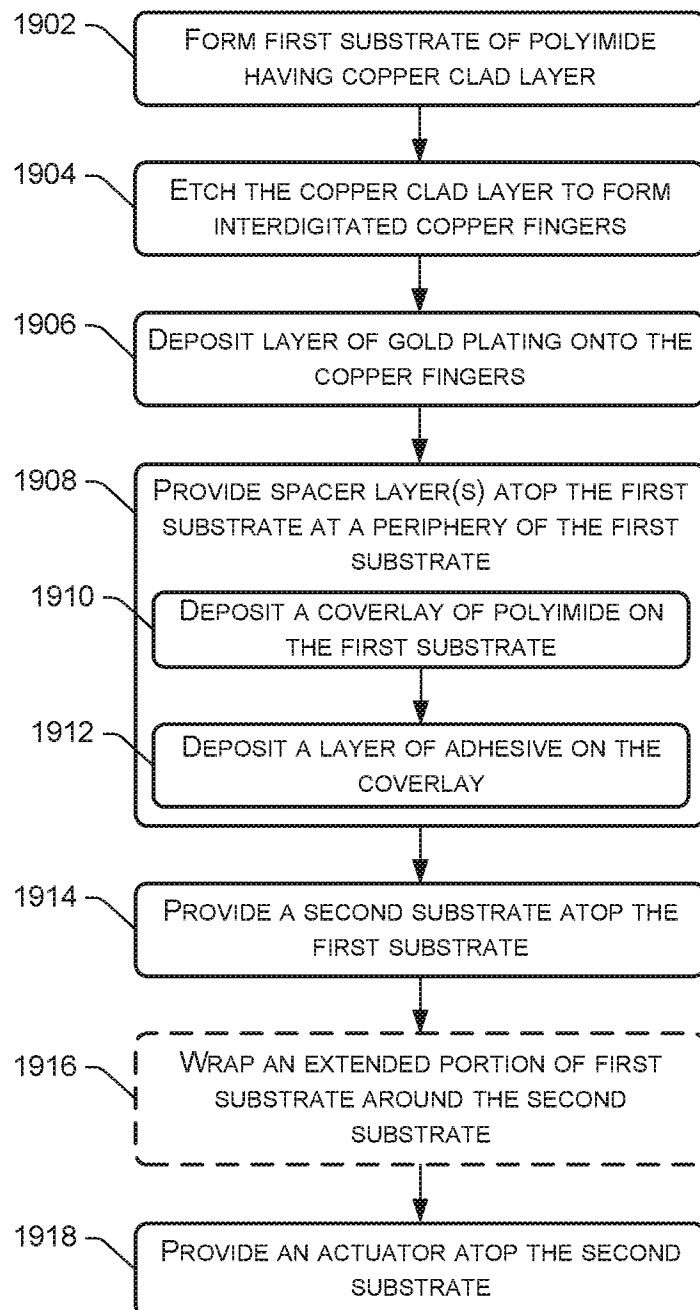


FIG. 18D

1900 ↘

**FIG. 19**

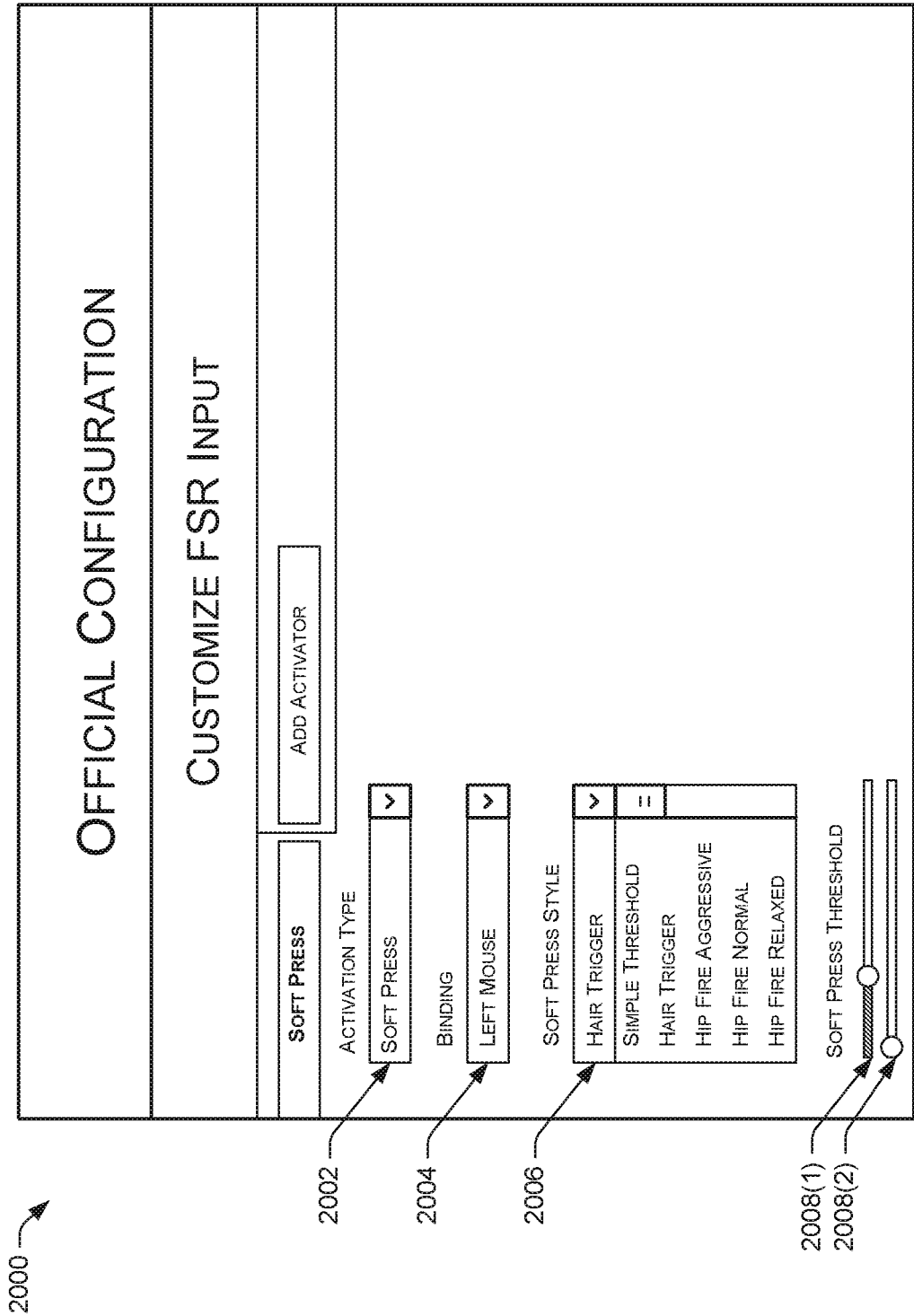


FIG. 20

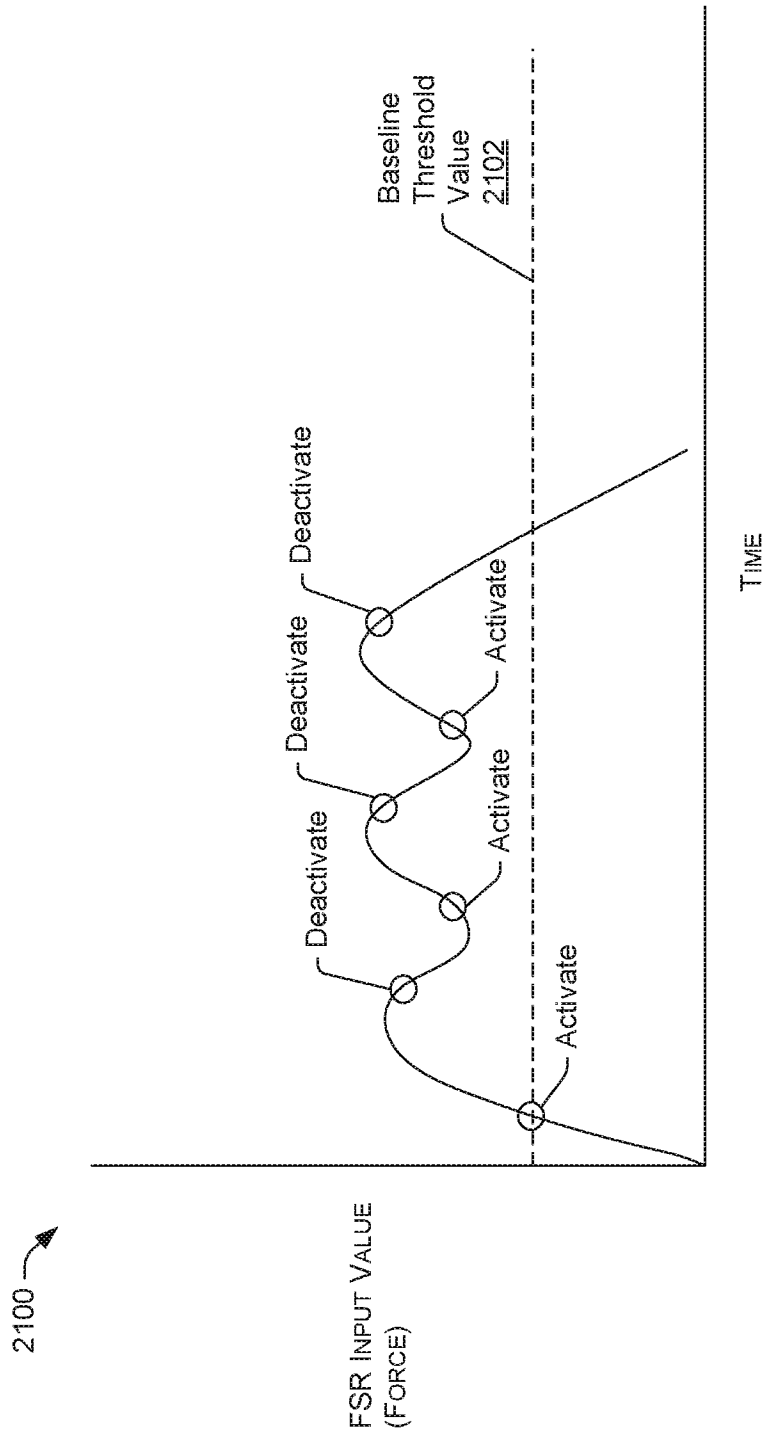


FIG. 21

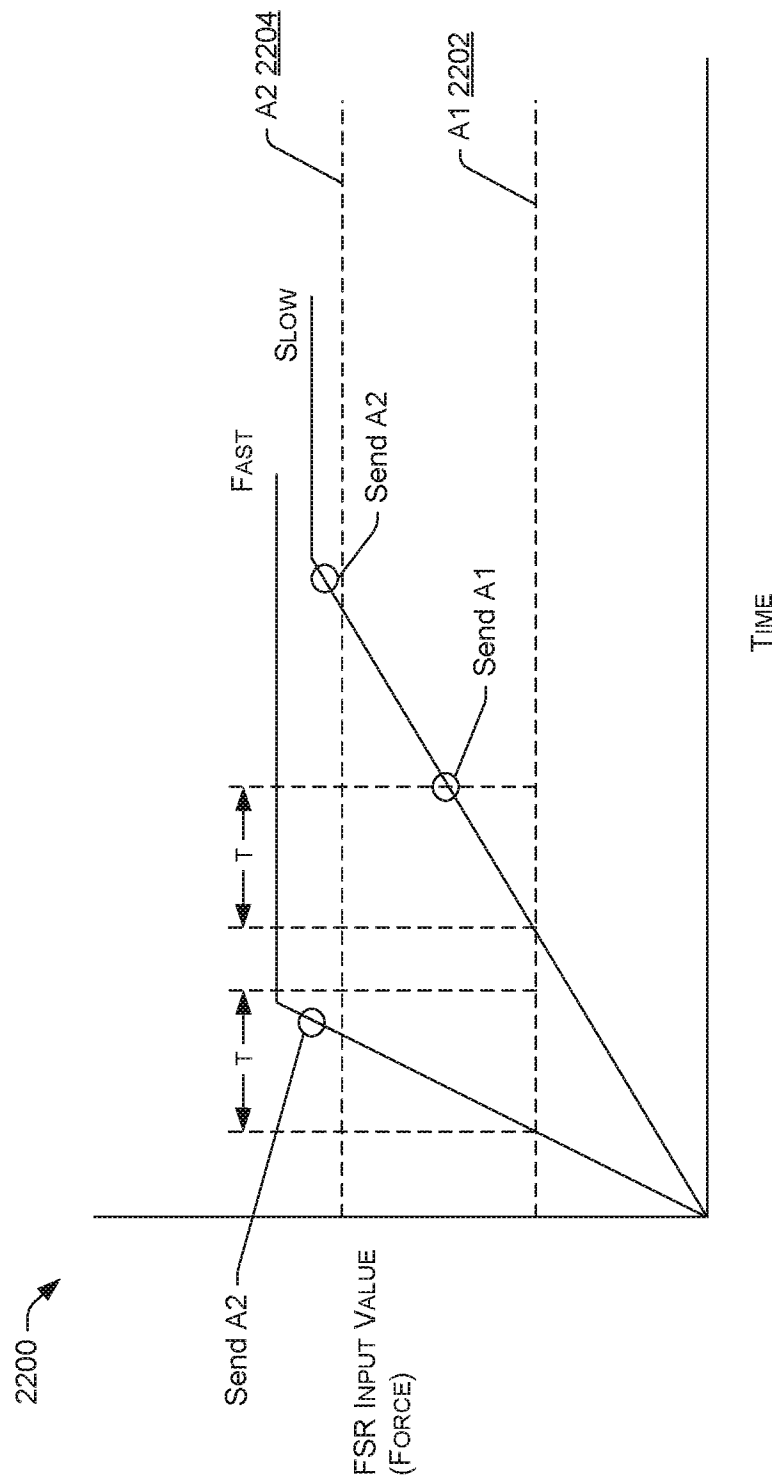


FIG. 22

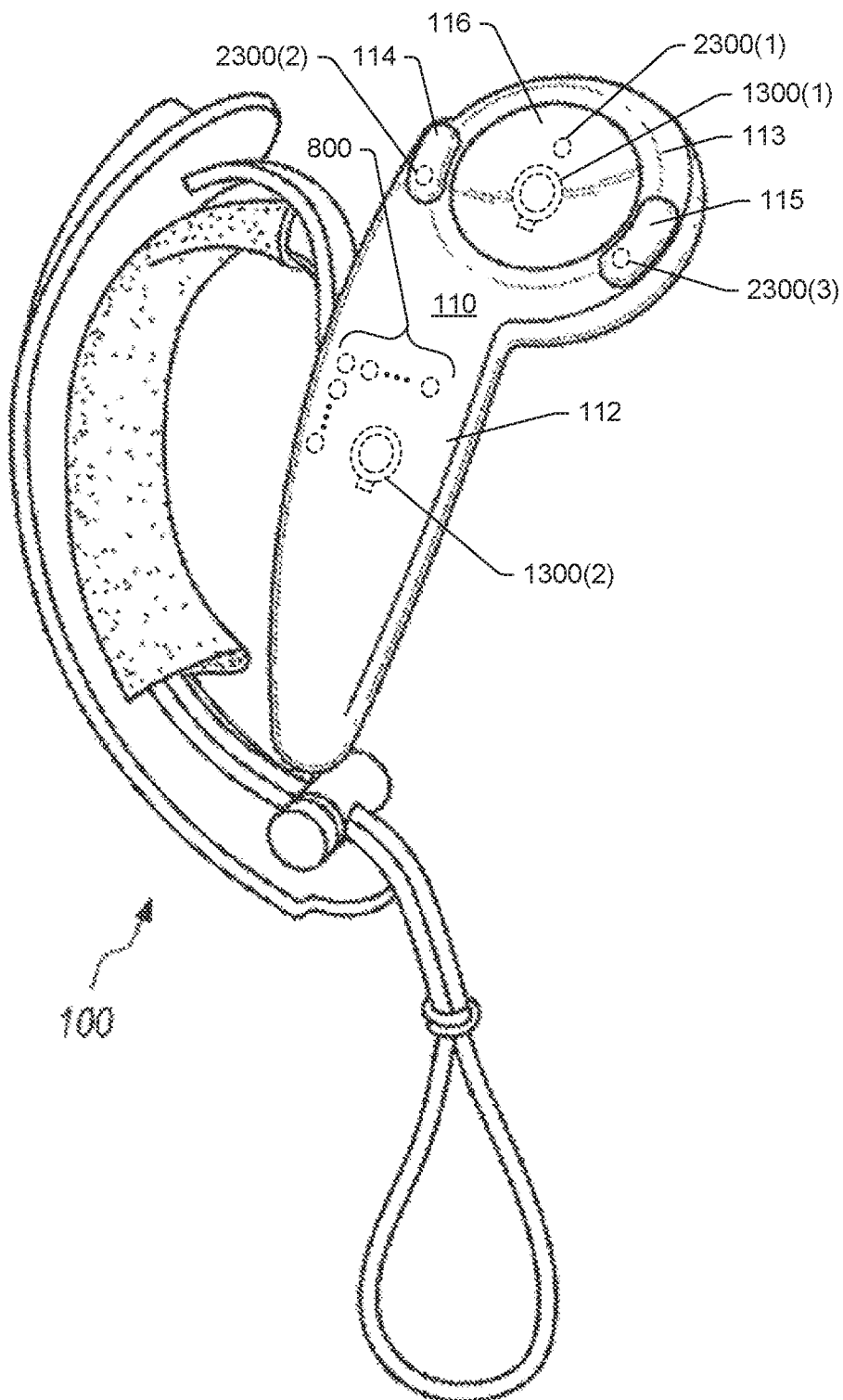
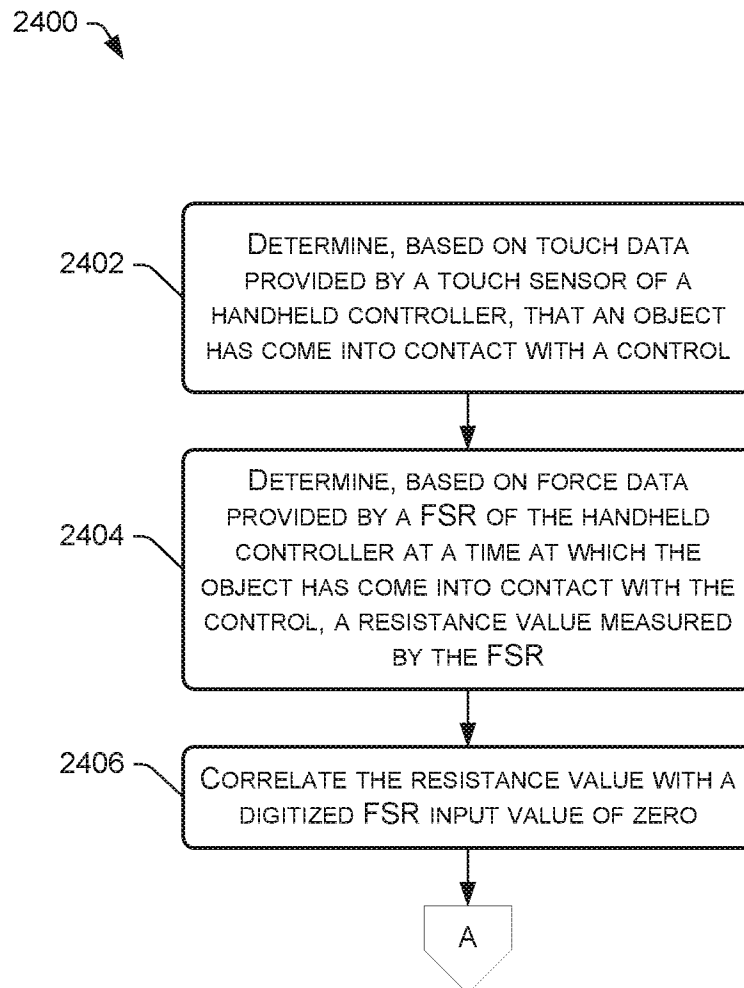
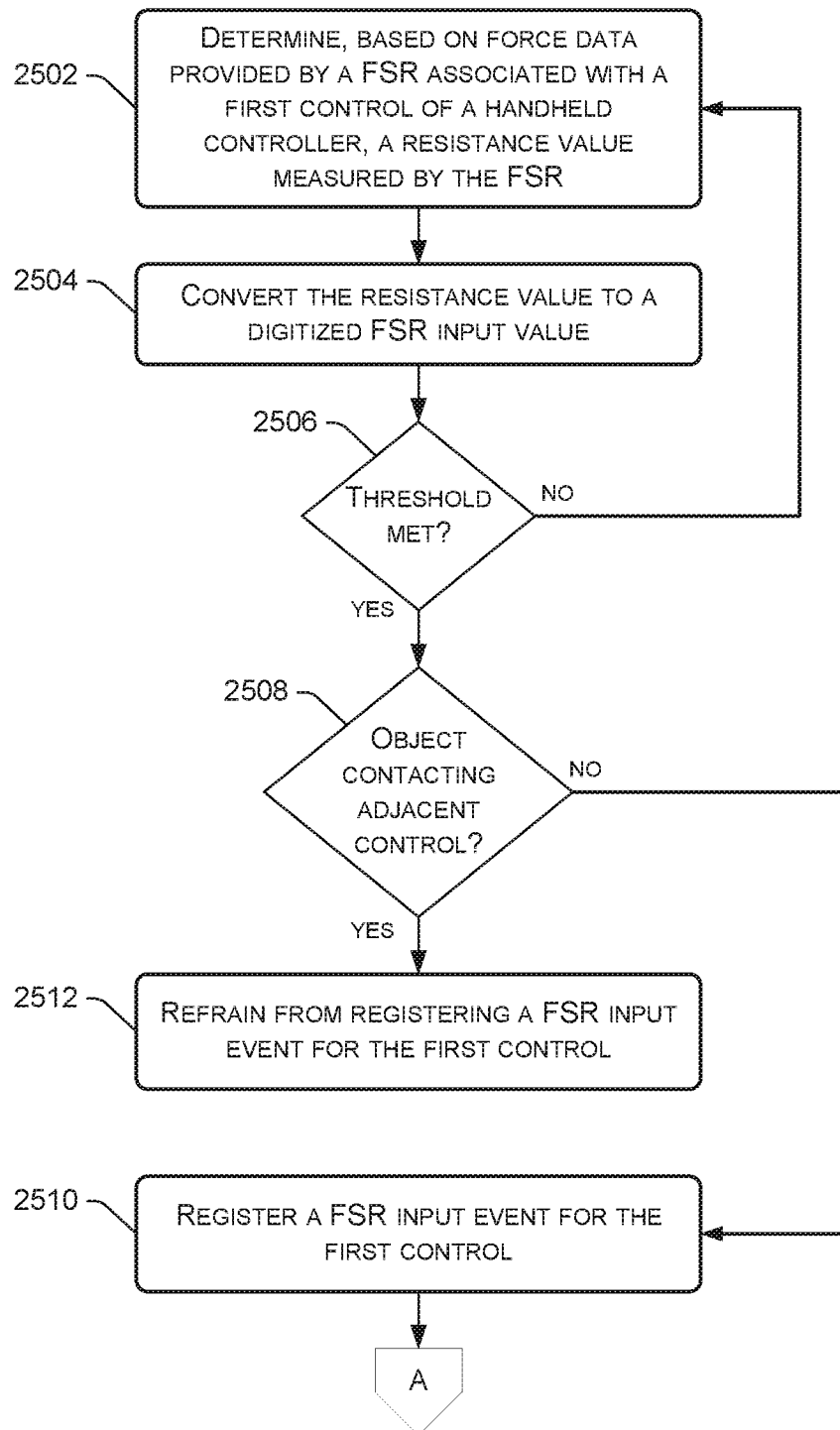
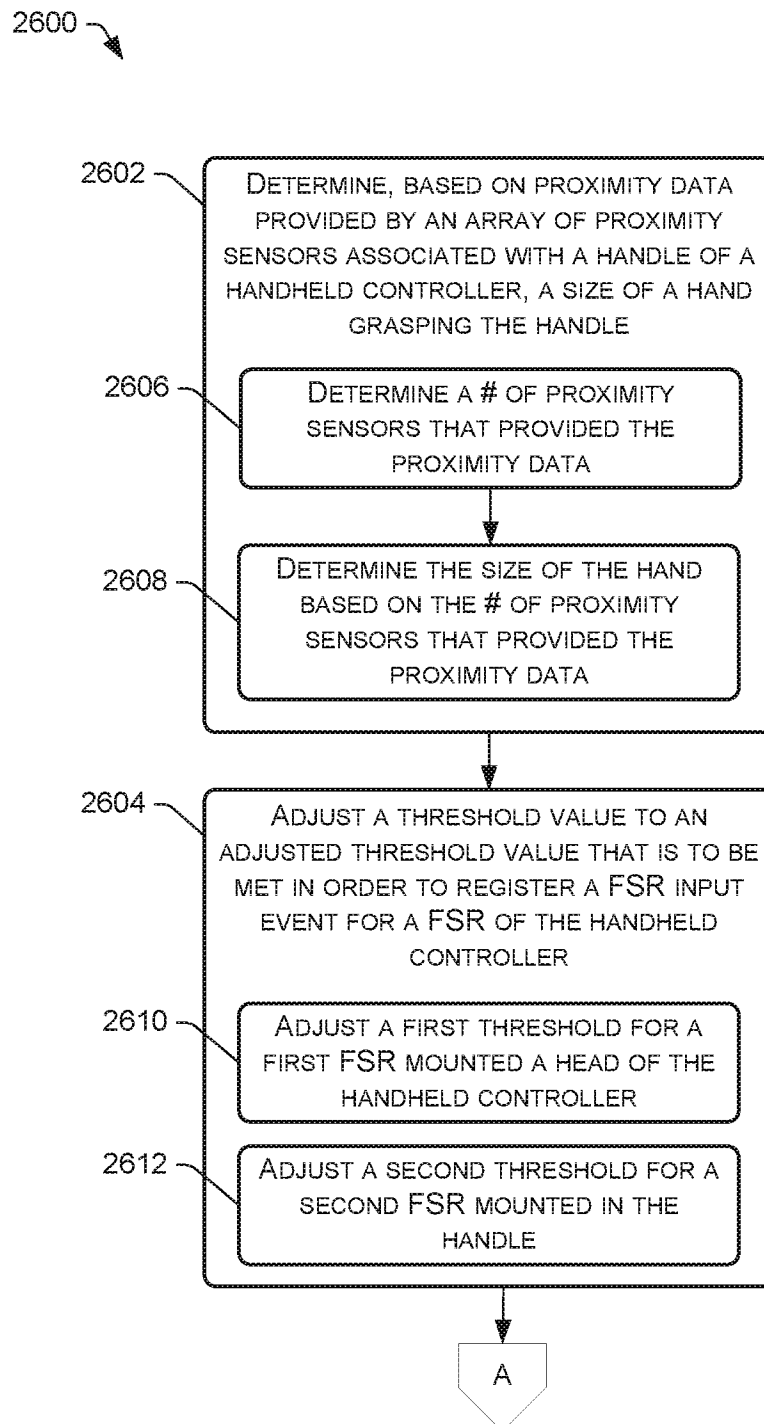


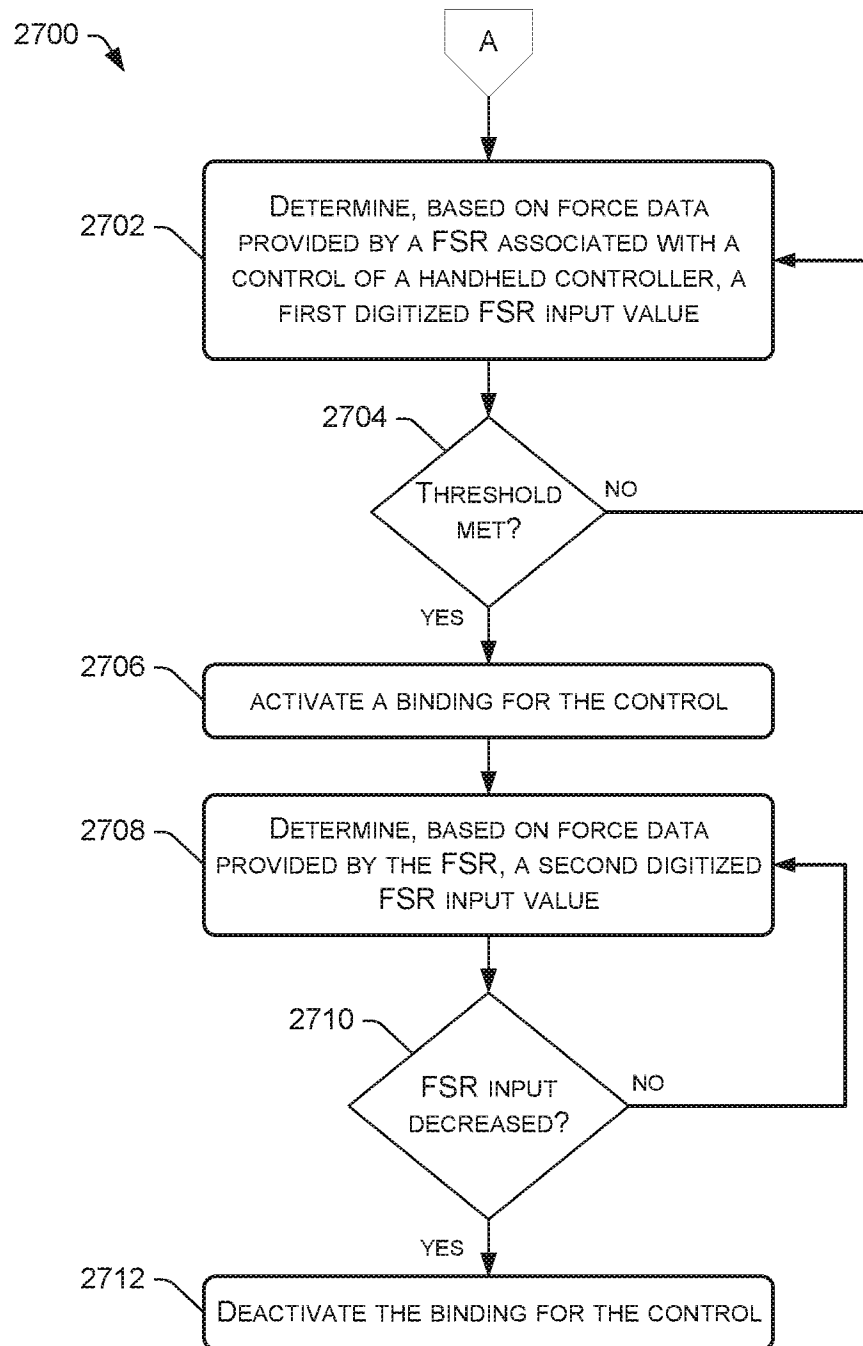
FIG. 23

*FIG. 24*

2500

**FIG. 25**

**FIG. 26**

**FIG. 27**

27/41

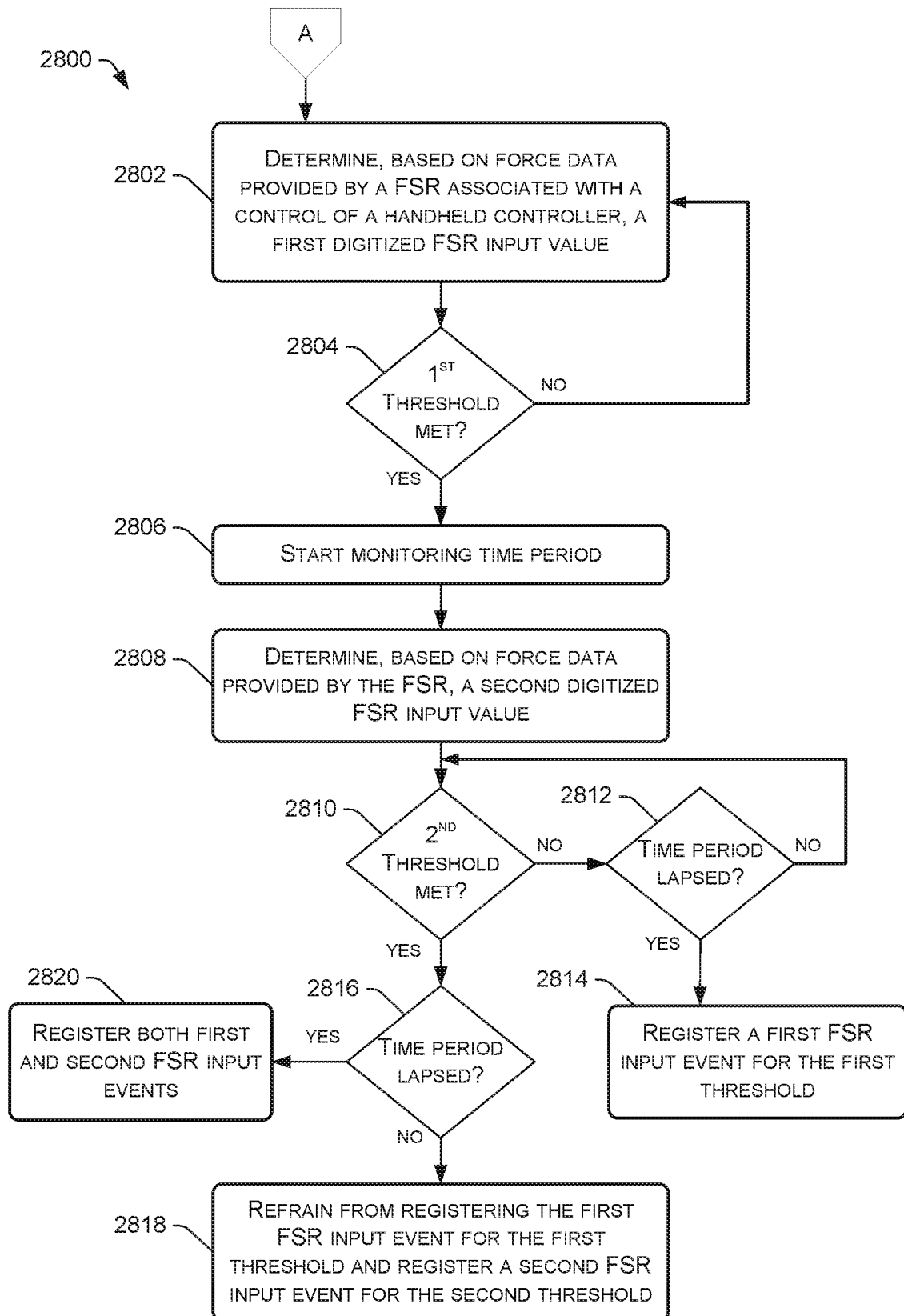
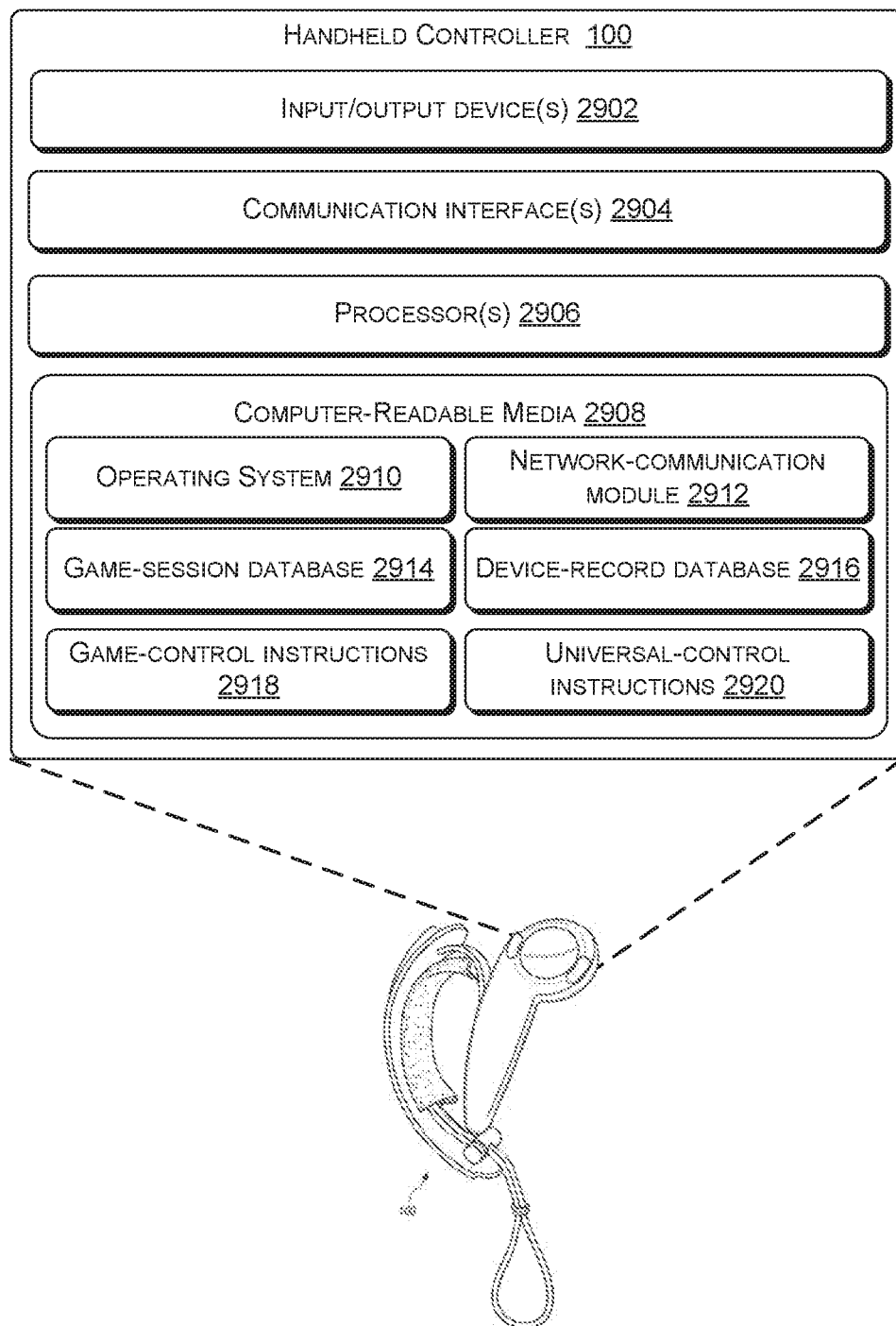


FIG. 28

**FIG. 29**

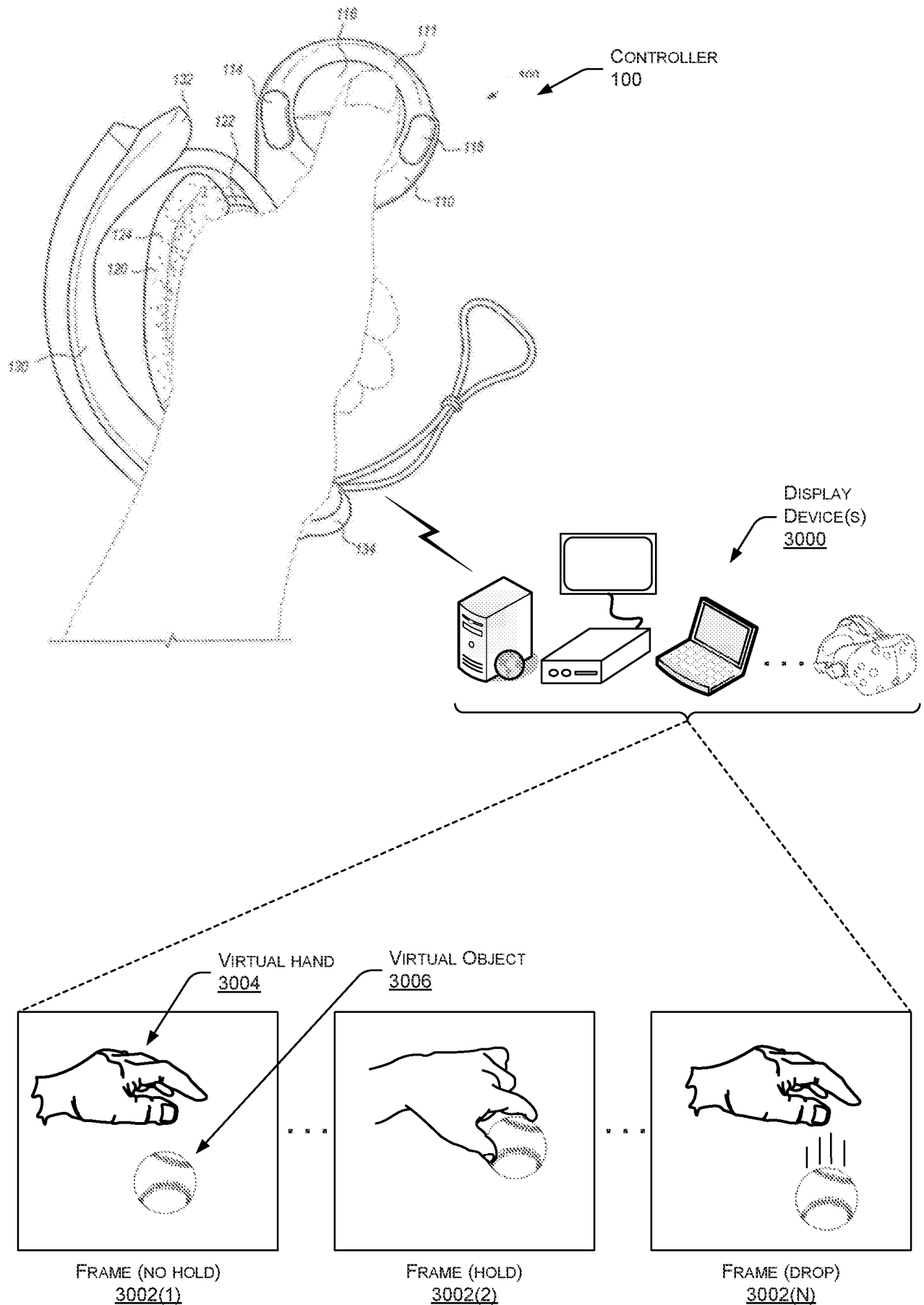


FIG. 30

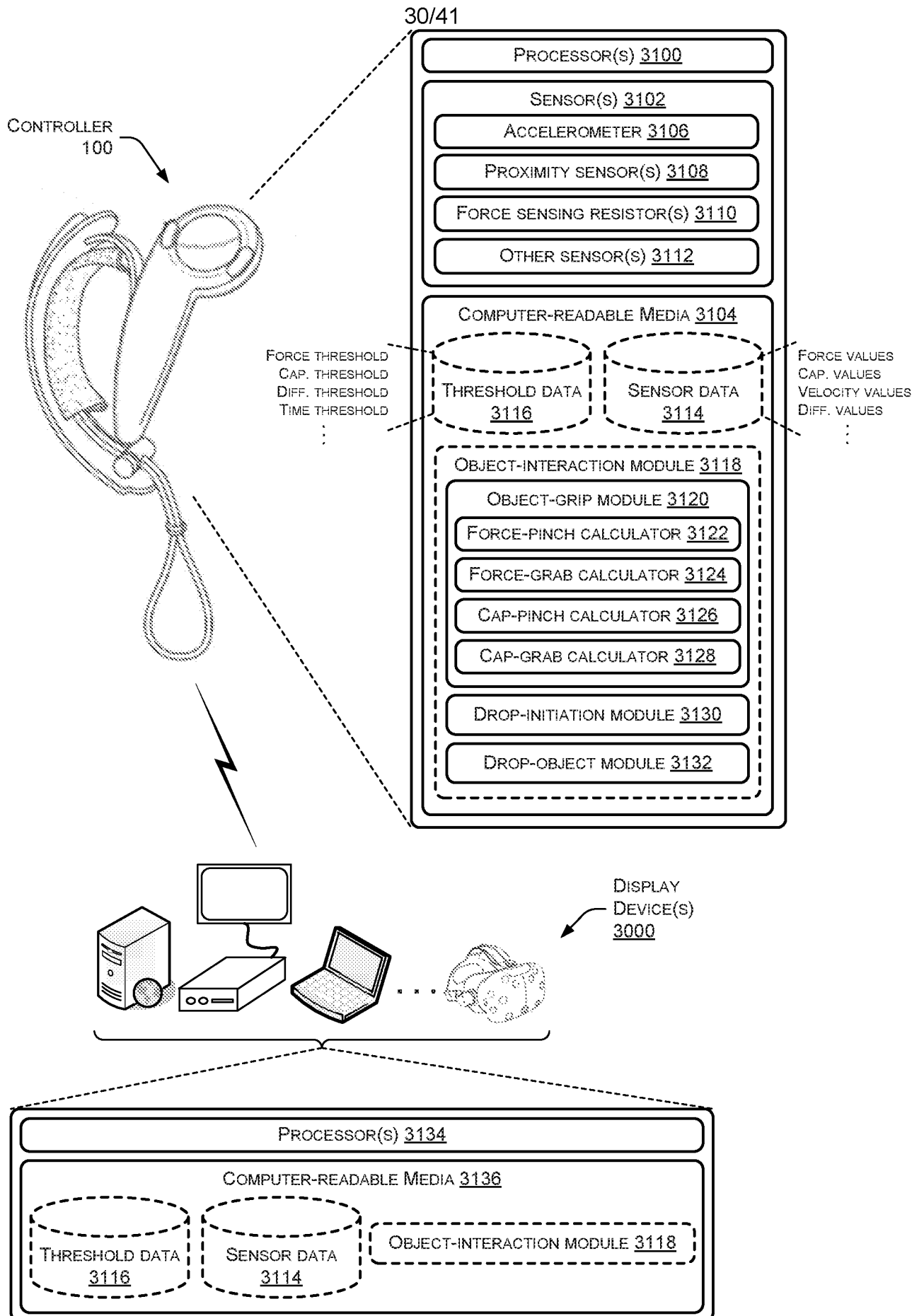
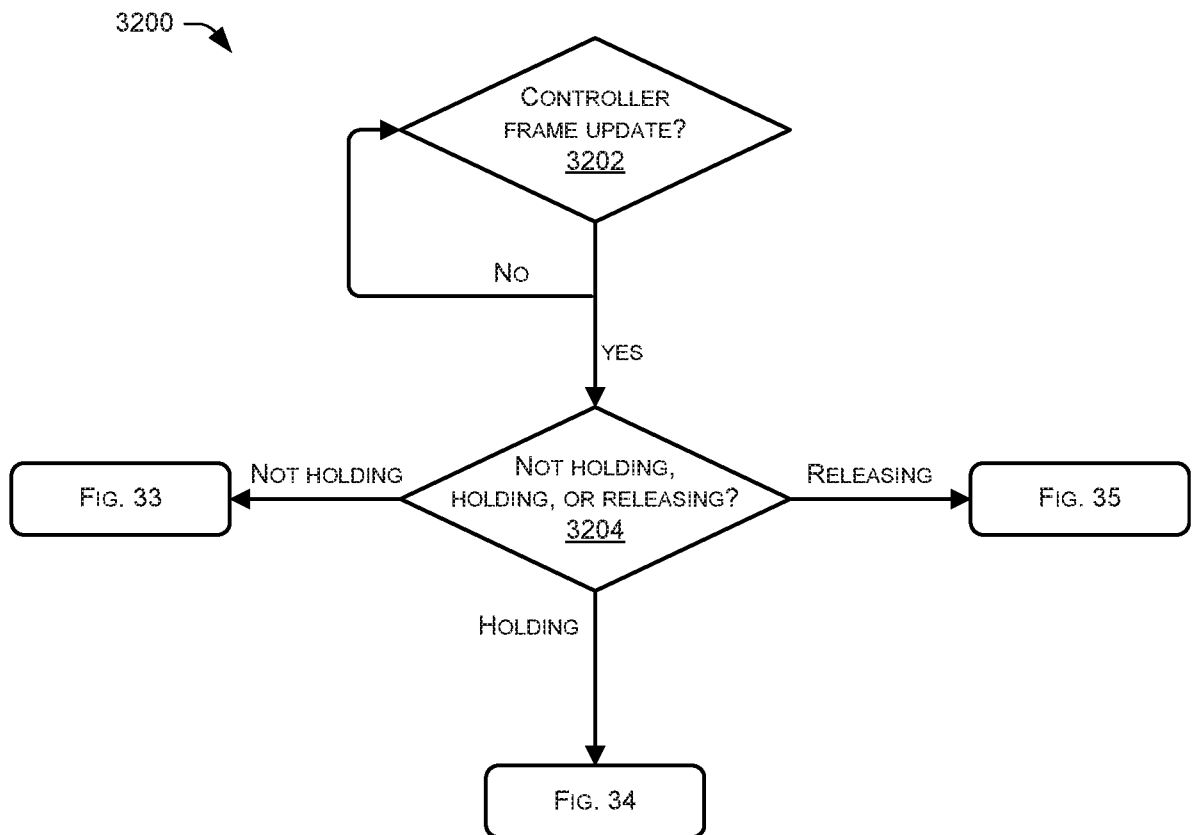


FIG. 31

**FIG. 32**

32/41

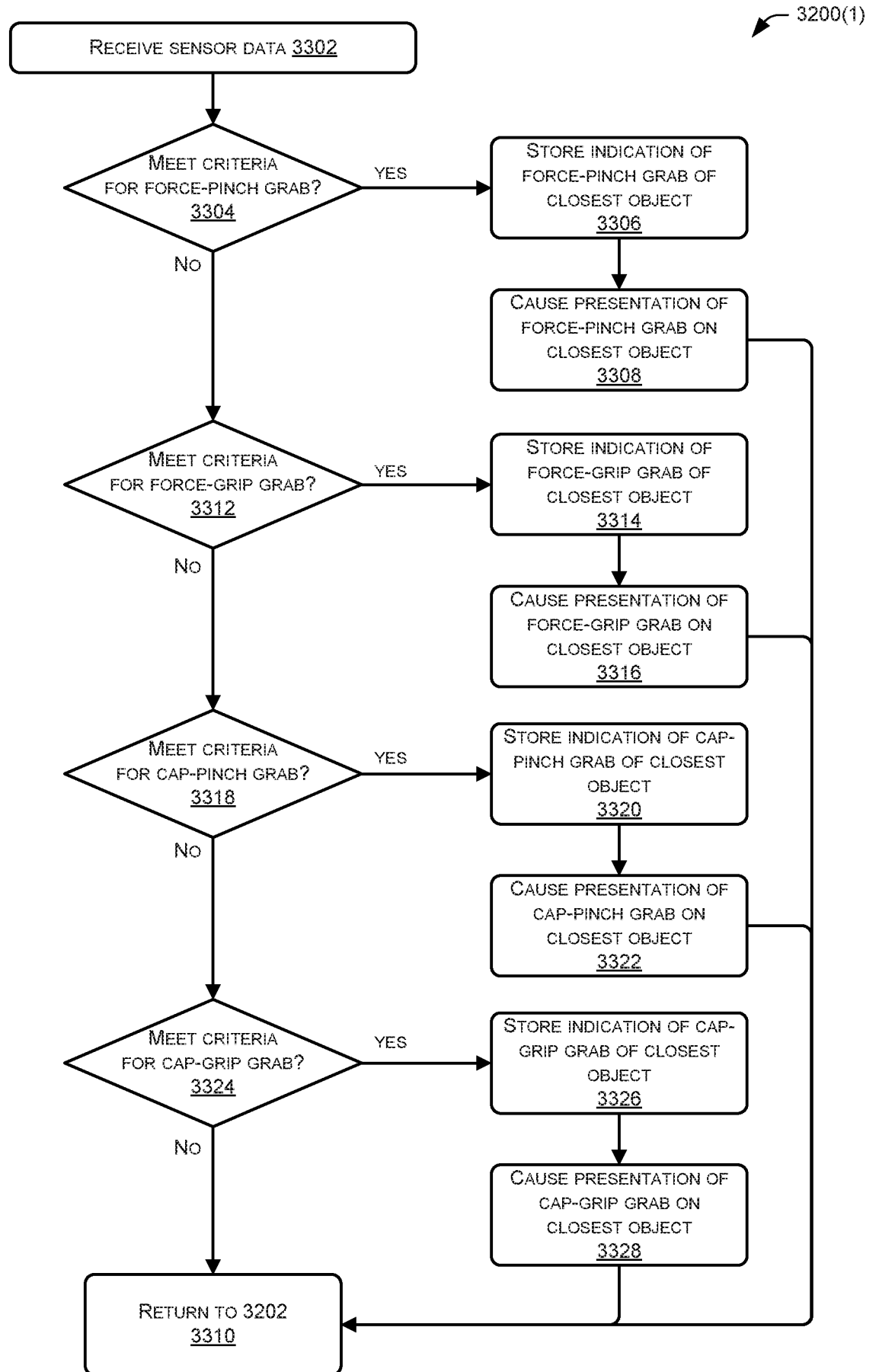


FIG. 33

33/41

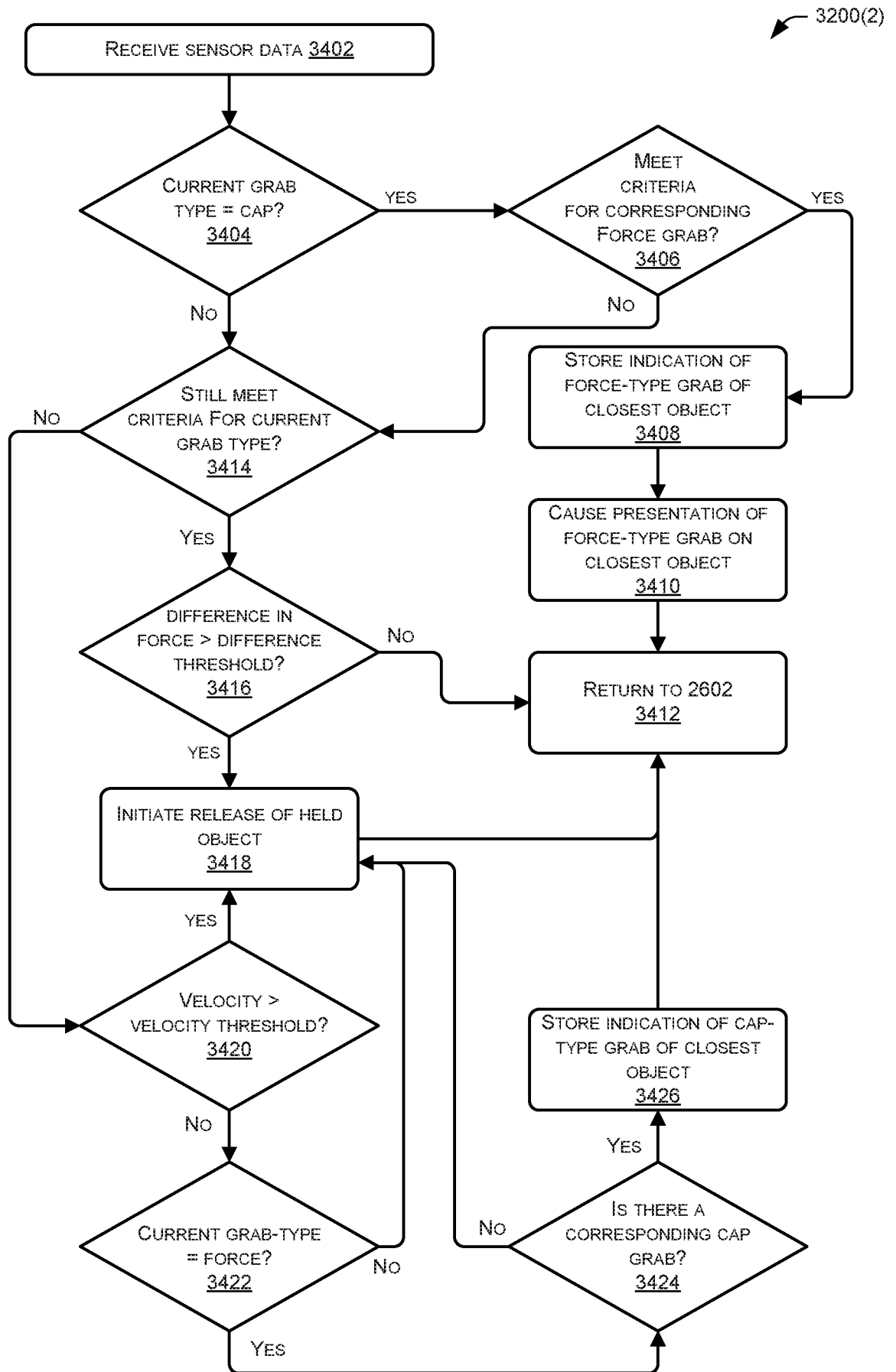


FIG. 34

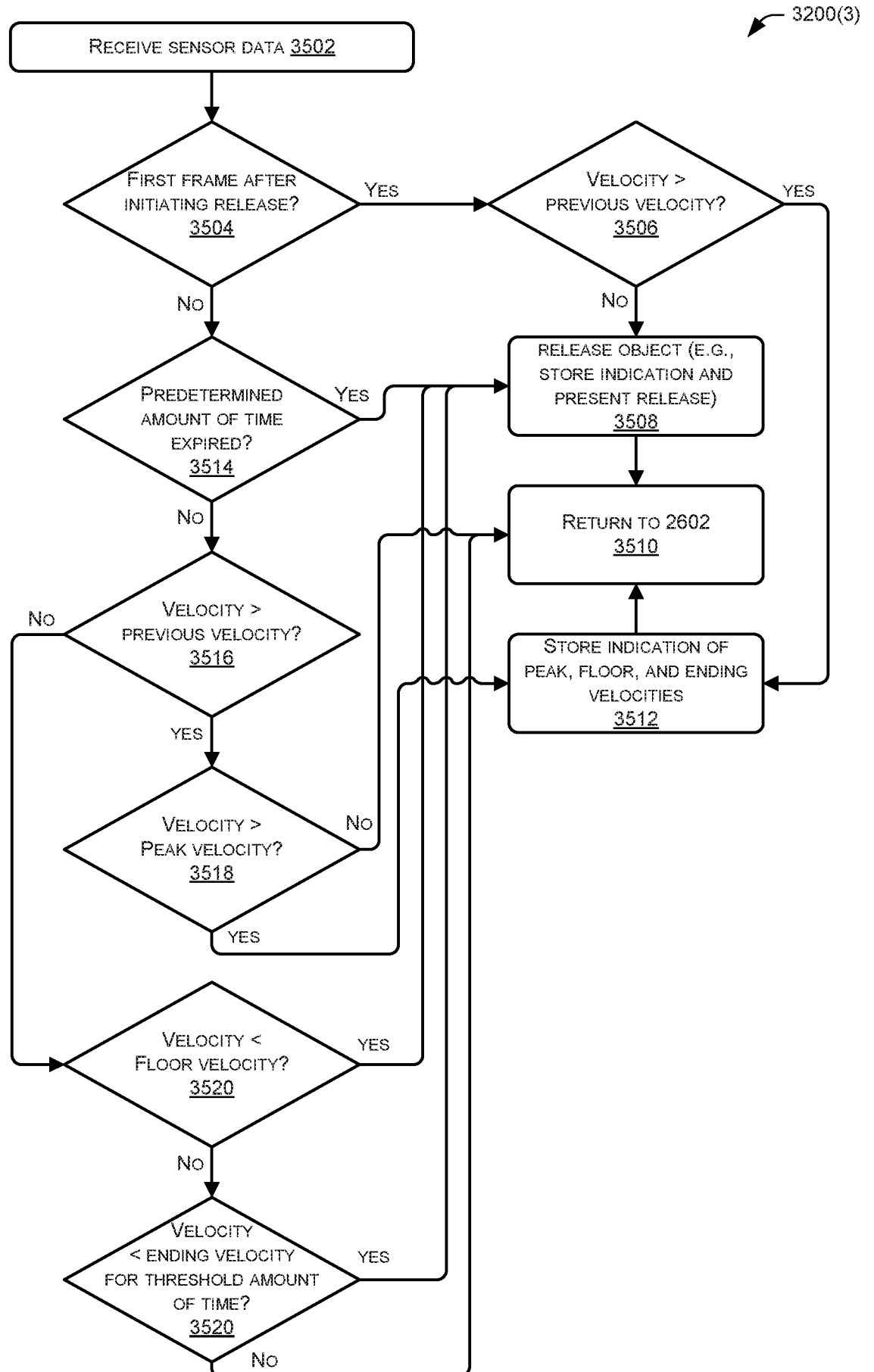
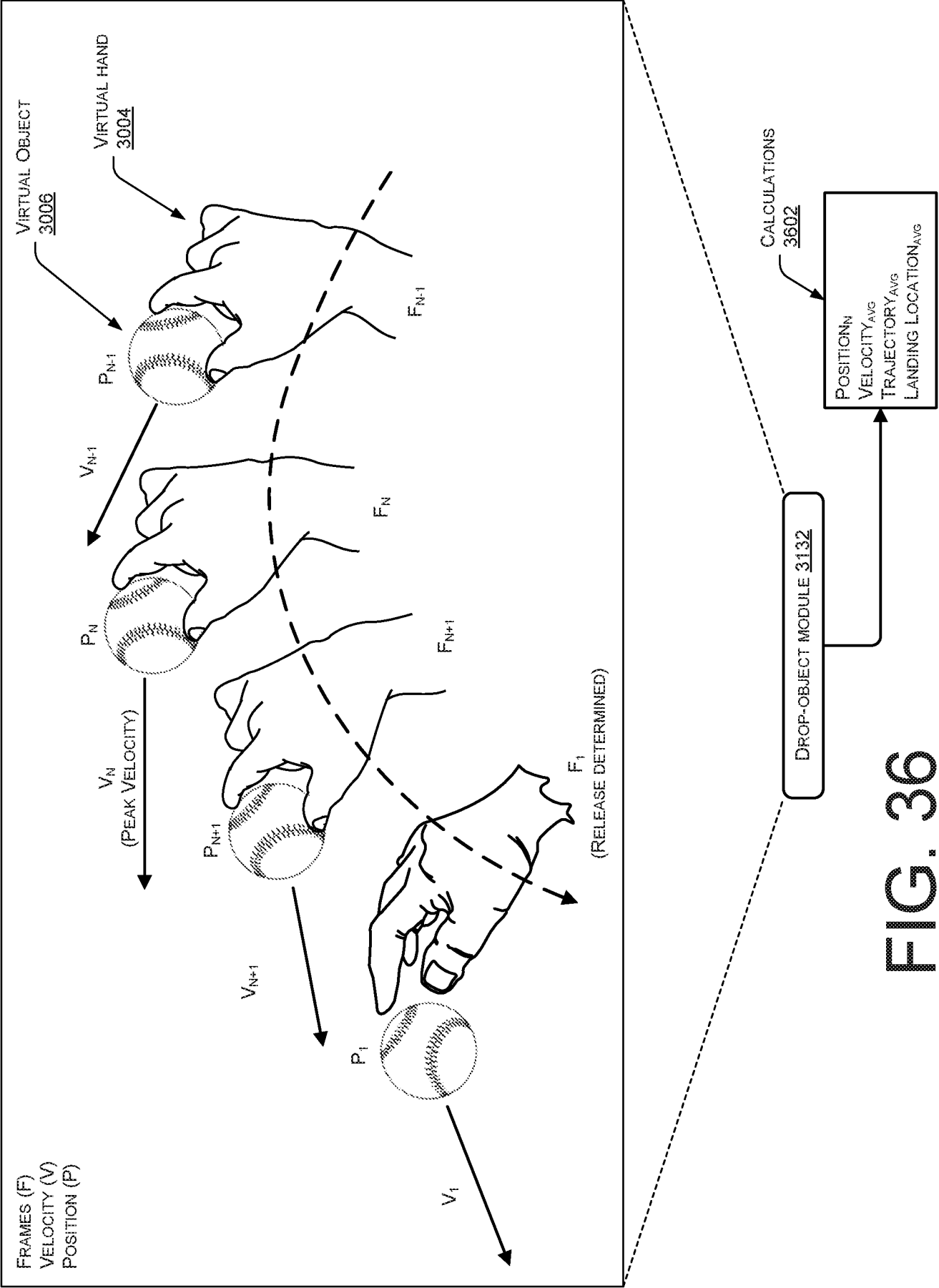


FIG. 35



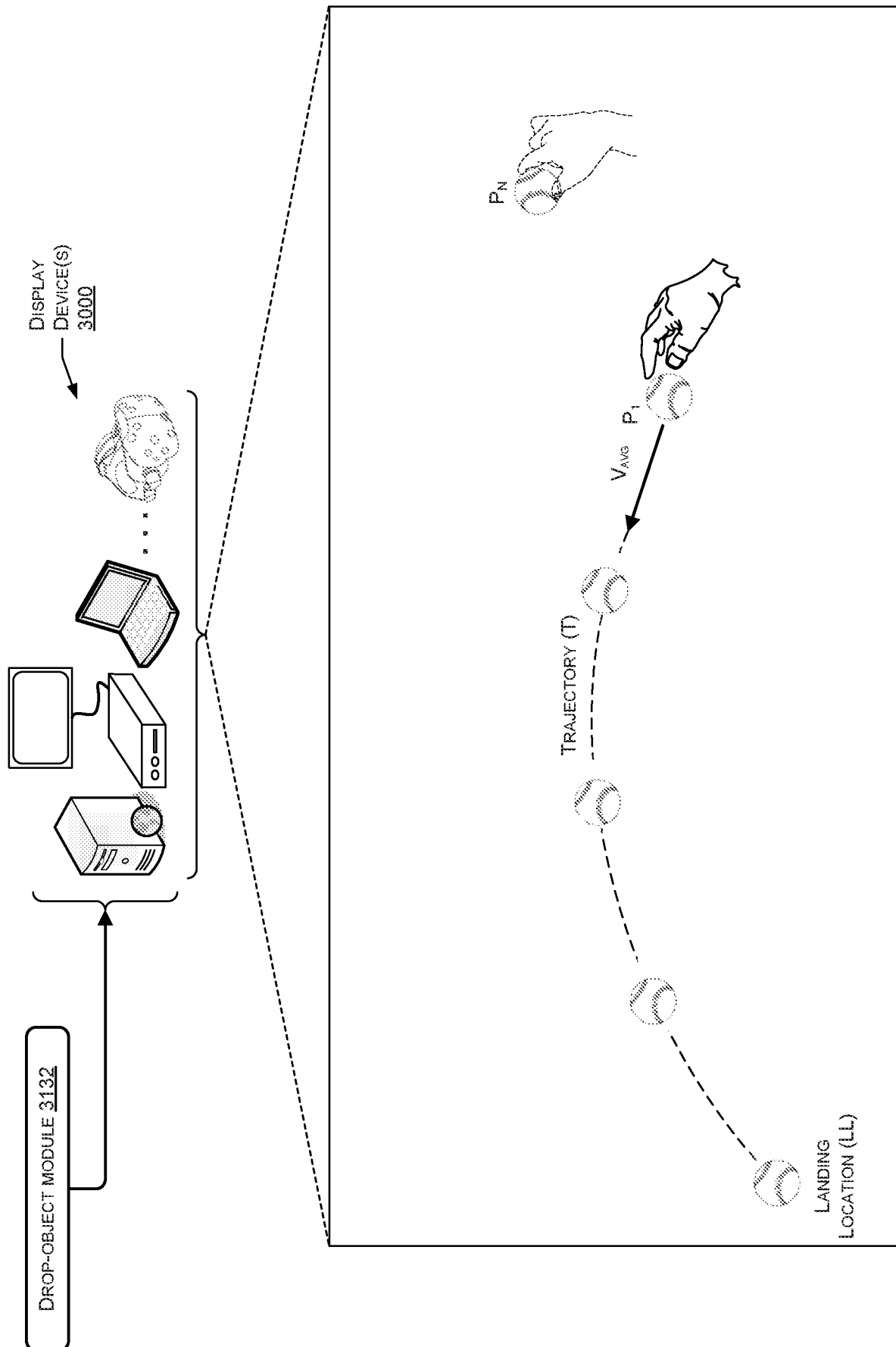


FIG. 37A

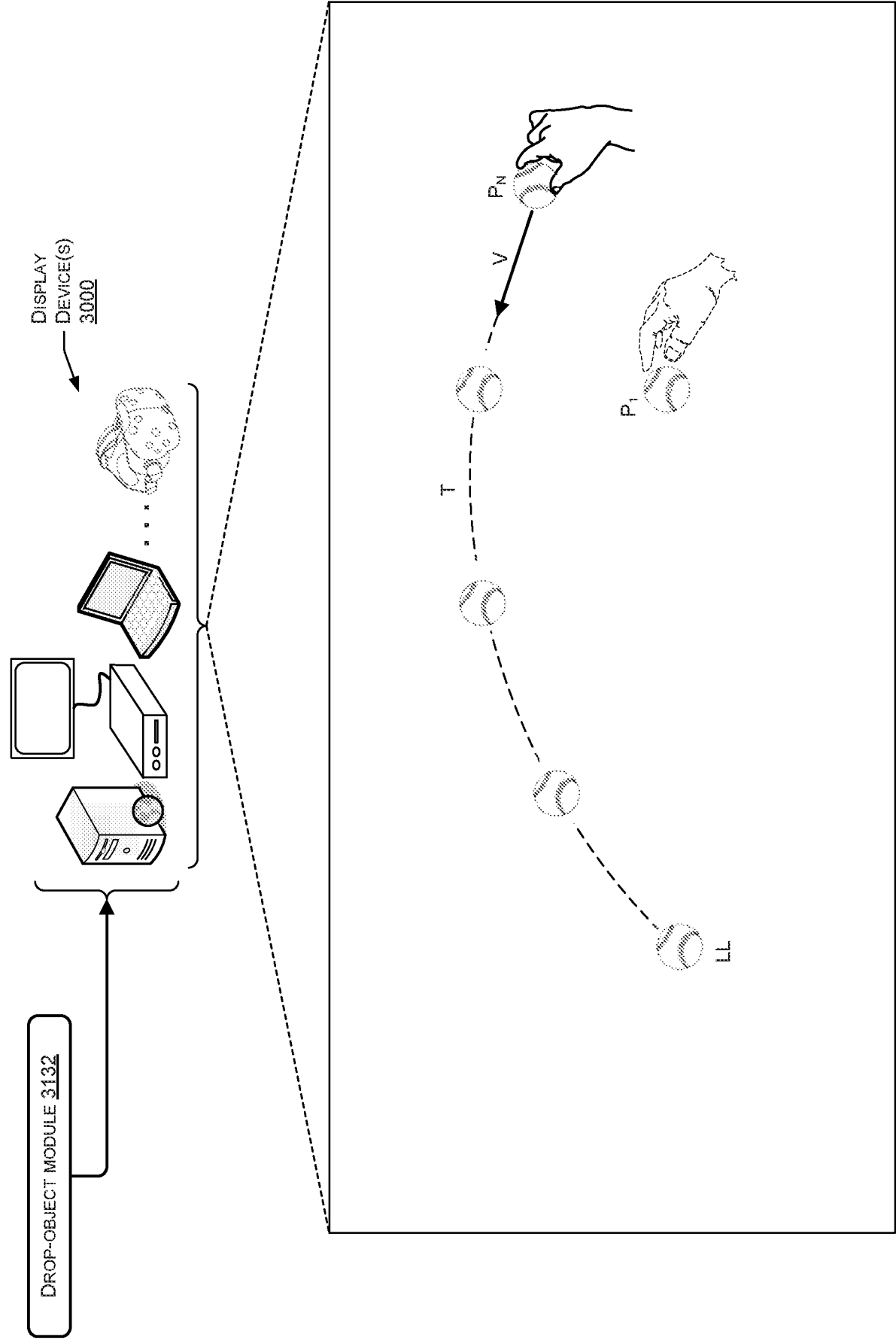


FIG. 37B

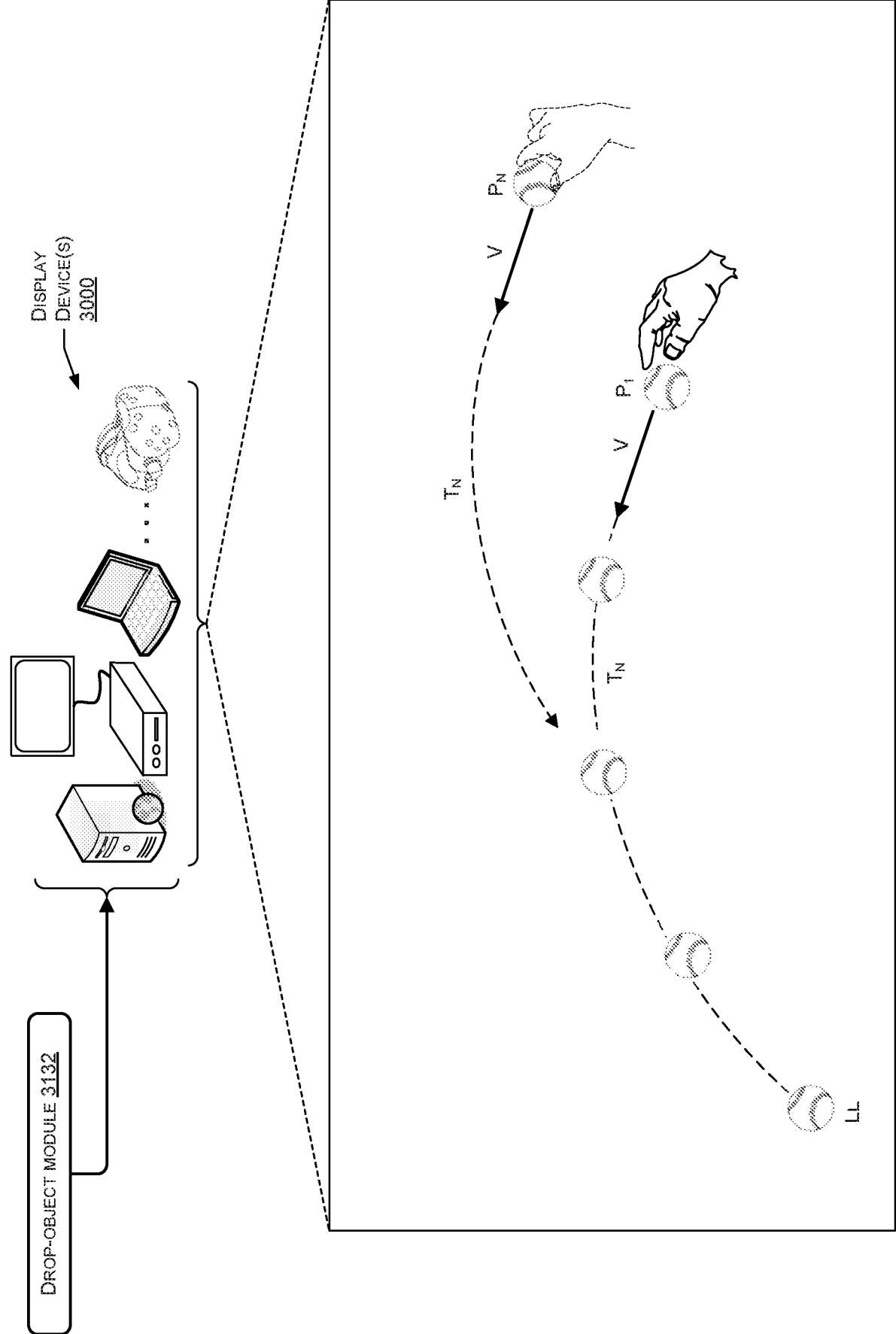


FIG. 37C

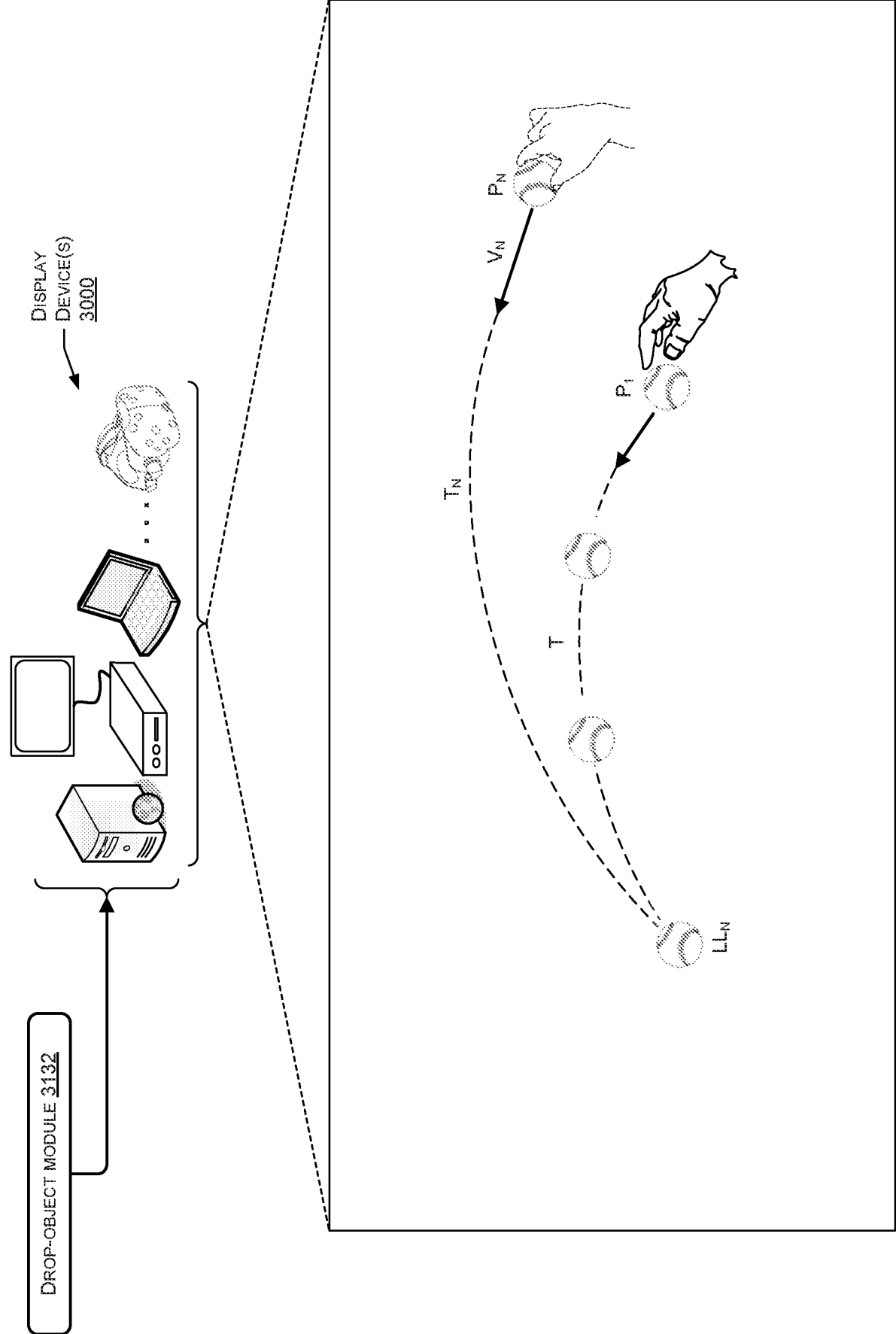


FIG. 37D

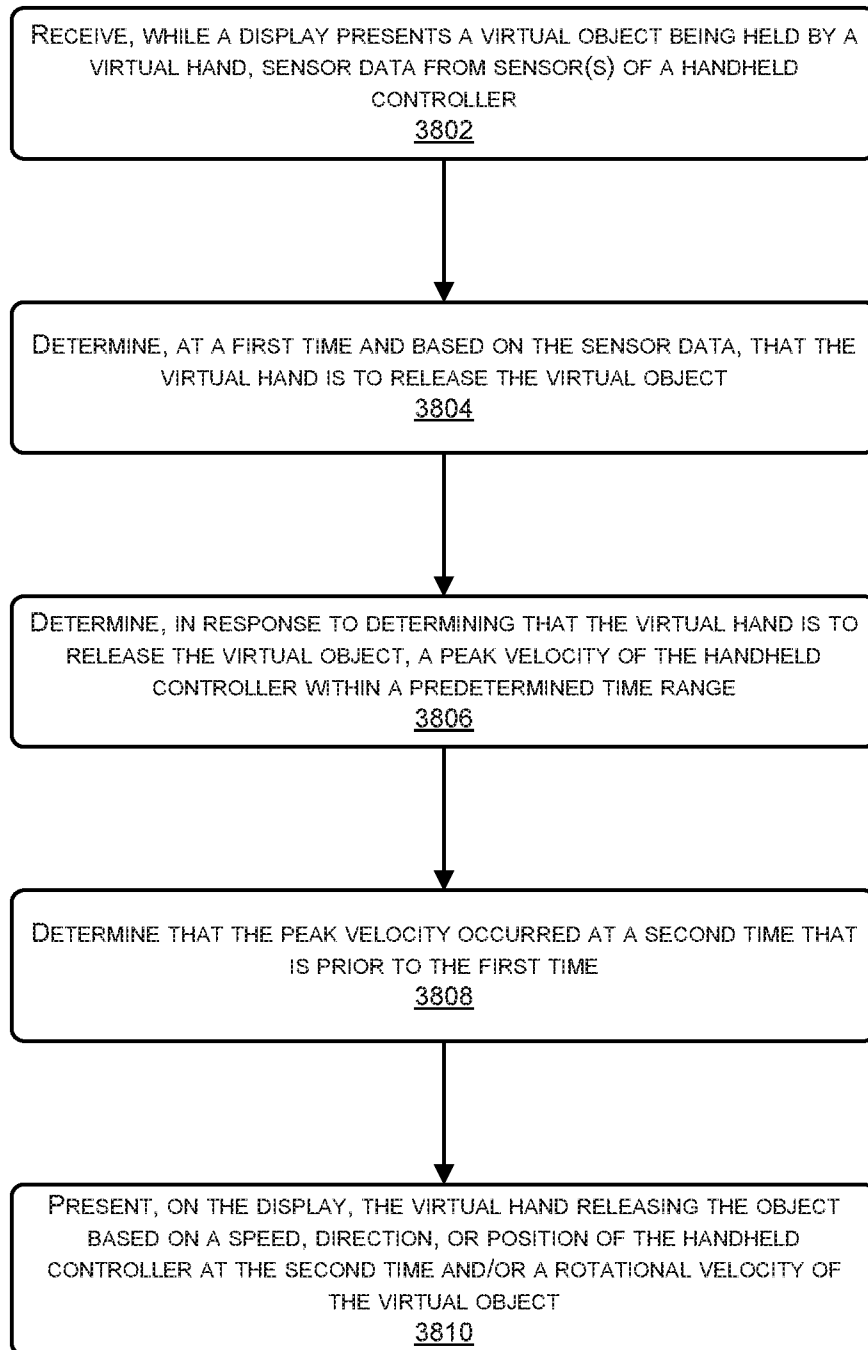


FIG. 38

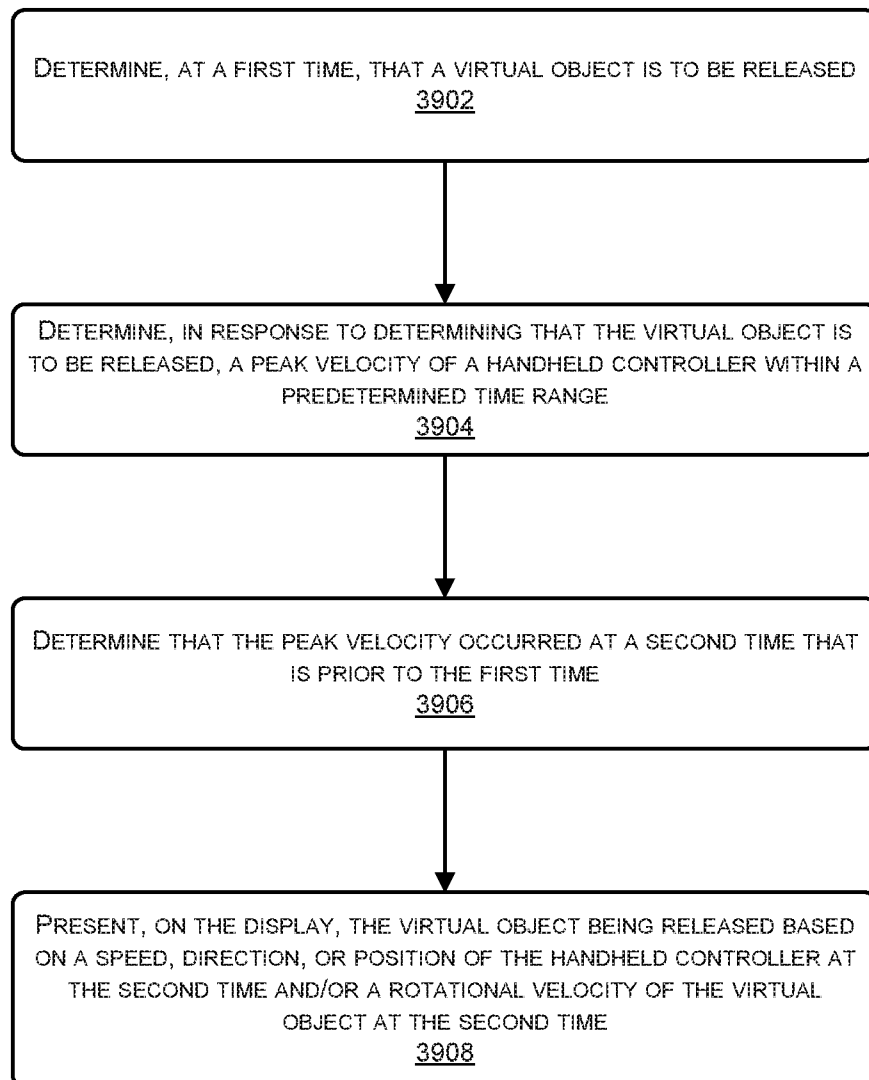


FIG. 39

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US19/37794

A. CLASSIFICATION OF SUBJECT MATTER

IPC - A63F 13/211, 13/218, 13/428; G06F 3/03; G06T 19/20 (2019.01)

CPC - A63F 13/211, 13/218, 13/428; G06F 3/0346, 3/0414; G06T 19/20

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

See Search History document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

See Search History document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y --- A	US 2017/0031503 A1 (SENSEL INC.) 02 February 2017; paragraphs [0034, 0048-0049, 0065-0070, 0081, 0088, 0104, 0110, 0130]	1-2, 4-5, 7-8, 12-15, 19-22, 26 ----- 3, 6 ----- 9-11, 16-18, 23-25
Y	US 2015/0091858 A1 (SENSEL INC.) 02 April 2015; figures 3, 7-9; figure 81; paragraphs [0147, 0153, 0156]	3, 6
A	US 2009/0005166 A1 (SATO, K.) 01 January 2009; figures 1, 6-8; paragraphs [0081-0089, 0096, 0108]	1-26

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

23 August 2019 (23.08.2019)

Date of mailing of the international search report

10 SEP 2019

Name and mailing address of the ISA/US

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents

P.O. Box 1450, Alexandria, Virginia 22313-1450

Facsimile No. 571-273-8300

Authorized officer

Shane Thomas

Telephone No. PCT Helpdesk: 571-272-4300