US 20220281123A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2022/0281123 A1

Robbins et al. (43) Pub. Date: Sep. 8, 2022

(54) SOFT-RIGID ROBOTIC JOINTS CONTROLLED BY DEEP REINFORCEMENT-LEARNING

(71) Applicant: **The Regents of the University of California**, Oakland, CA (US)

(72) Inventors: **Ash Robbins**, Santa Cruz, CA (US); **Mircea Teodorescu**, Santa Cruz, CA (US)

(21) Appl. No.: **17/684,489**

(22) Filed: **Mar. 2, 2022**

Related U.S. Application Data

(60) Provisional application No. 63/155,490, filed on Mar. 2, 2021, provisional application No. 63/217,854, filed on Jul. 2, 2021.

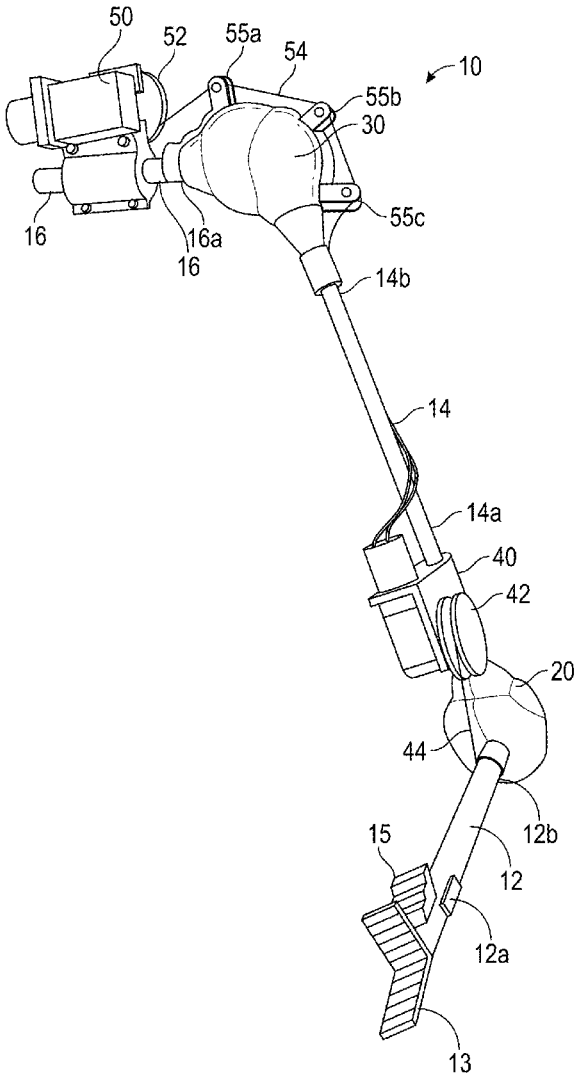
Publication Classification

(51) **Int. Cl.**
B25J 17/00 (2006.01)
B25J 9/16 (2006.01)

(52) **U.S. Cl.**
CPC ***B25J 17/00*** (2013.01); ***B25J 9/1671*** (2013.01); ***B25J 9/163*** (2013.01)

(57) **ABSTRACT**

A robotic arm having one or more hybrid (soft-rigid) joints includes a first link, a second link, and a joint interconnecting the first link and the second link, such that the first link is movable relative to the second link along an axis of motion. The joint includes: a socket component coupled to a distal end portion of the second link and a ball component coupled to a proximal end portion of first link, the ball component is configured to rotationally fit within the socket component. The joint also includes a flexible membrane encasing the socket component and the ball component. The robotic arm is controllable using a reinforcement learning algorithm training using a simulation of the robotic arm and optionally, further training in a physical world.



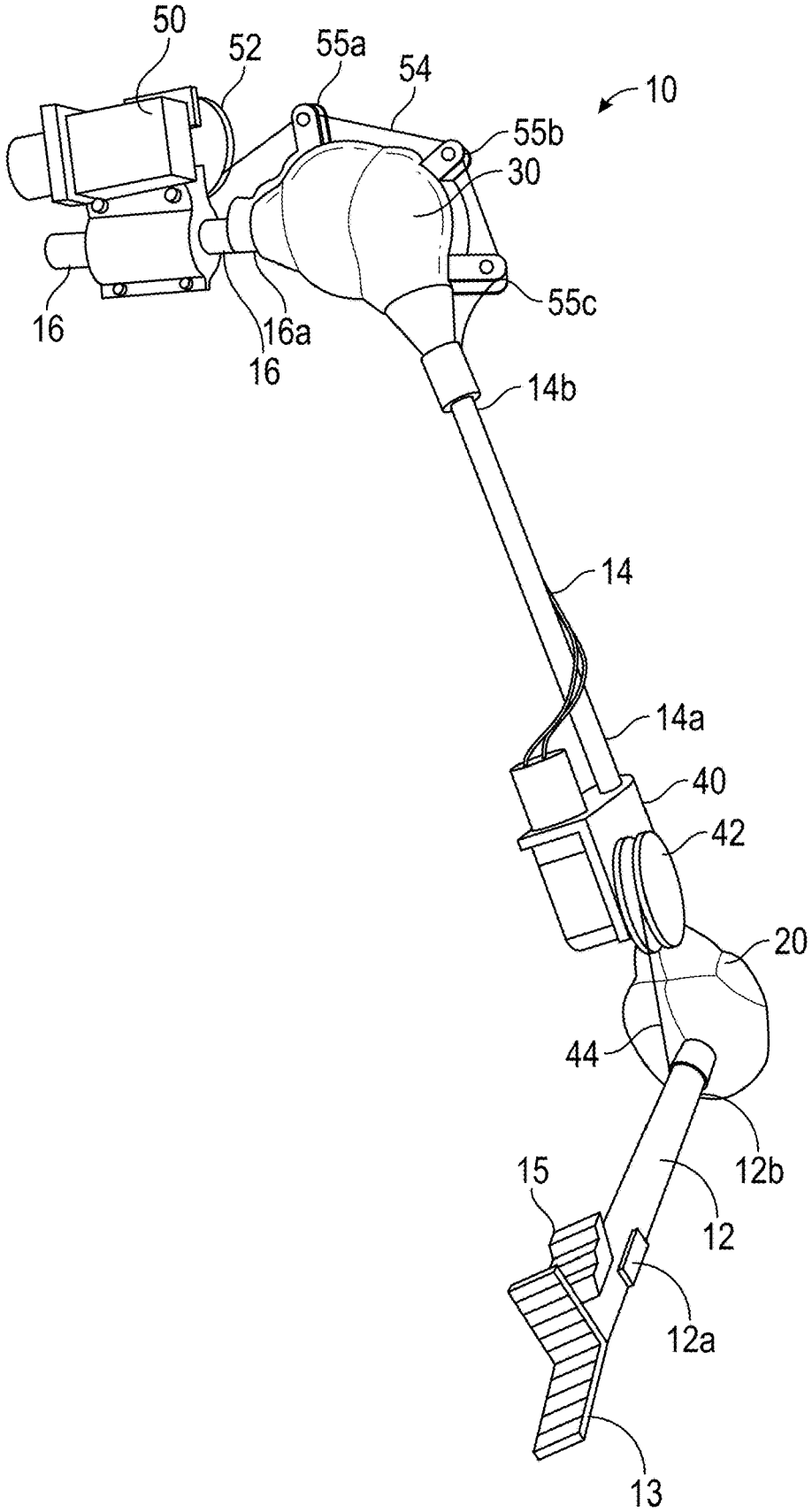

50
52
55a
54
10
55b
30
55c
16
16a
16
14b
14
14a
40
42
20
44
12b
15
12
12a
13

FIG. 1

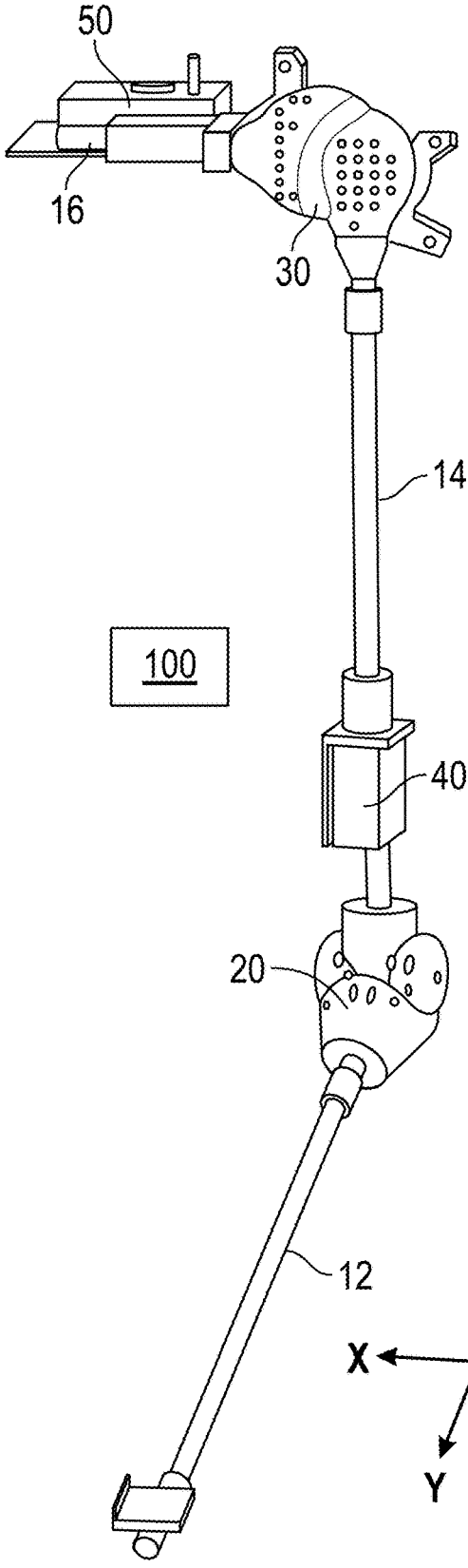


FIG. 2

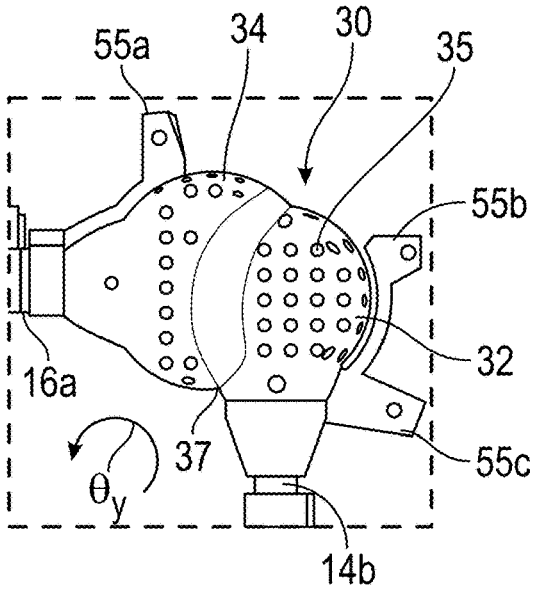


FIG. 3

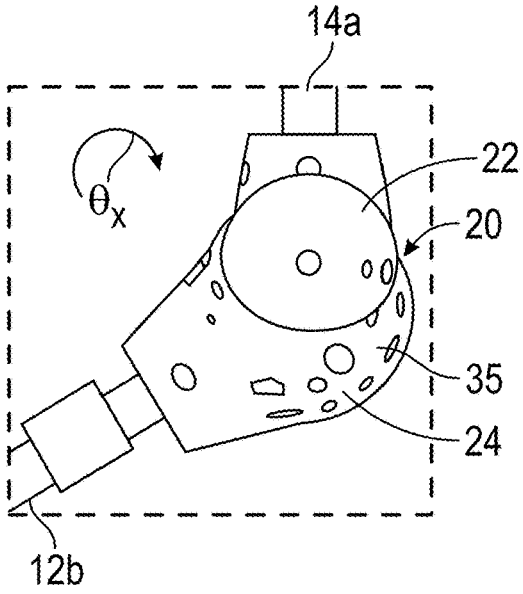


FIG. 4

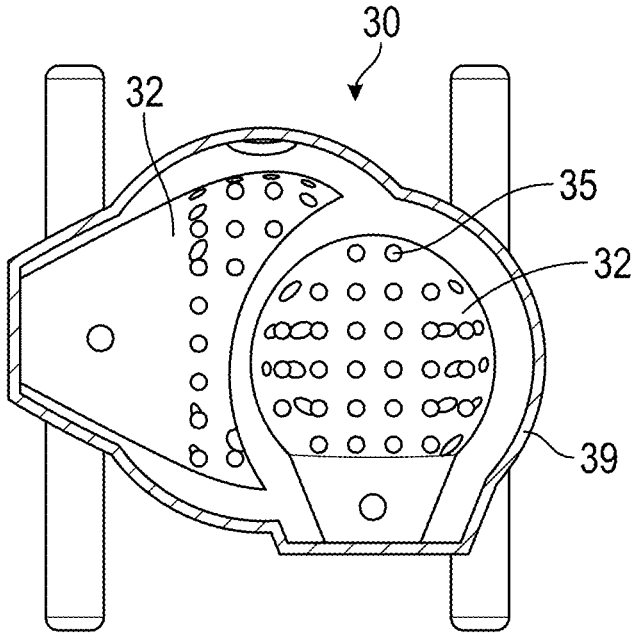


FIG. 5

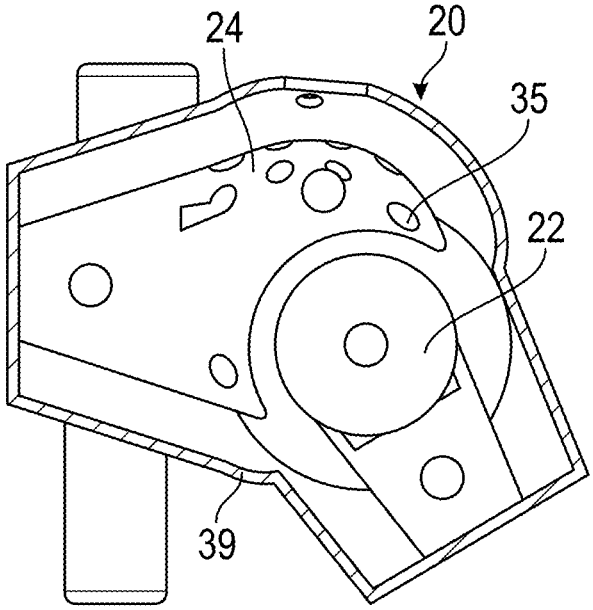


FIG. 6

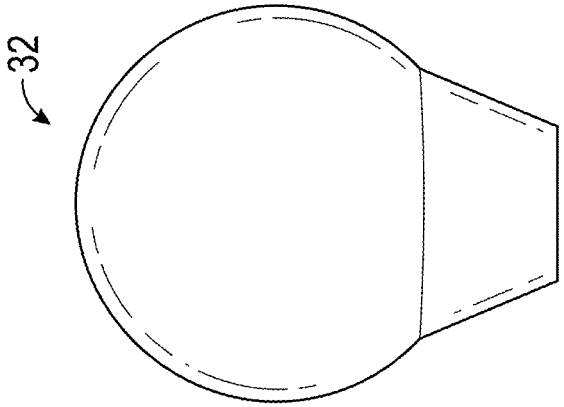


FIG. 9

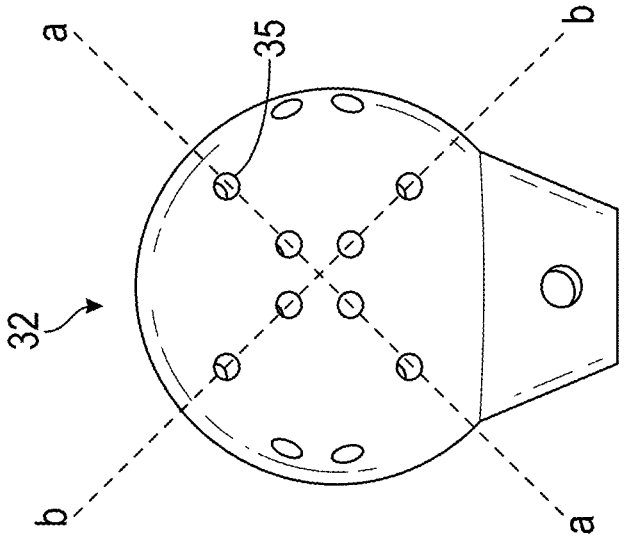


FIG. 8

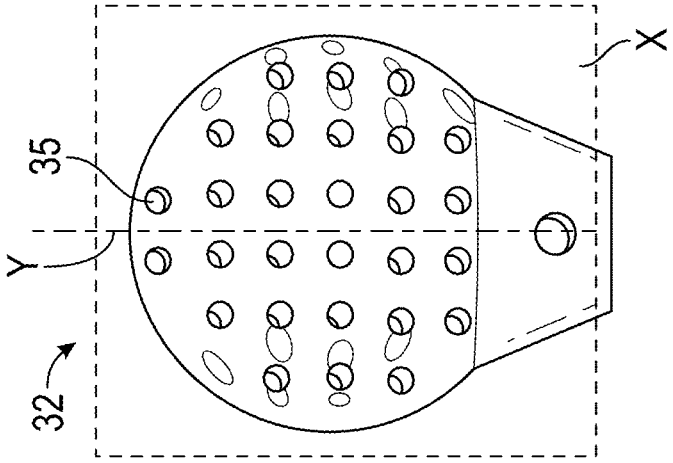


FIG. 7

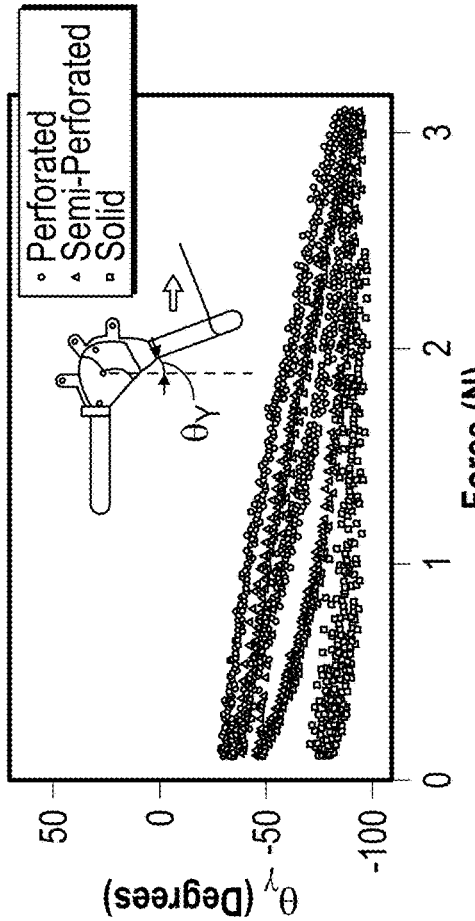


FIG. 11

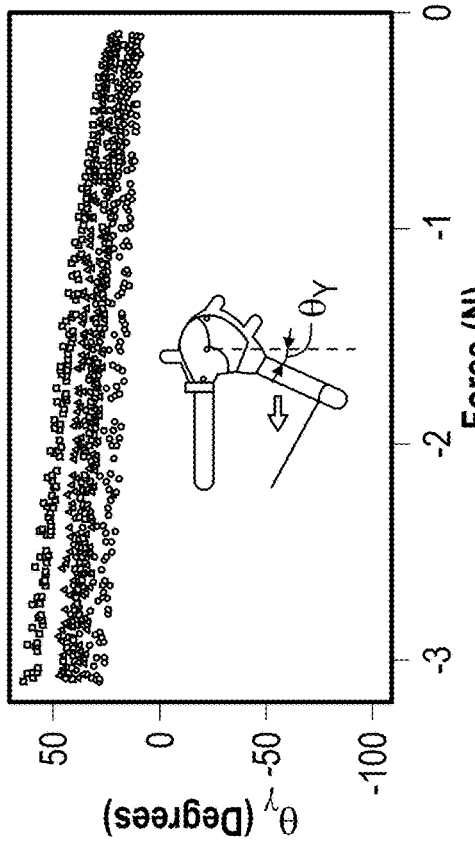


FIG. 10

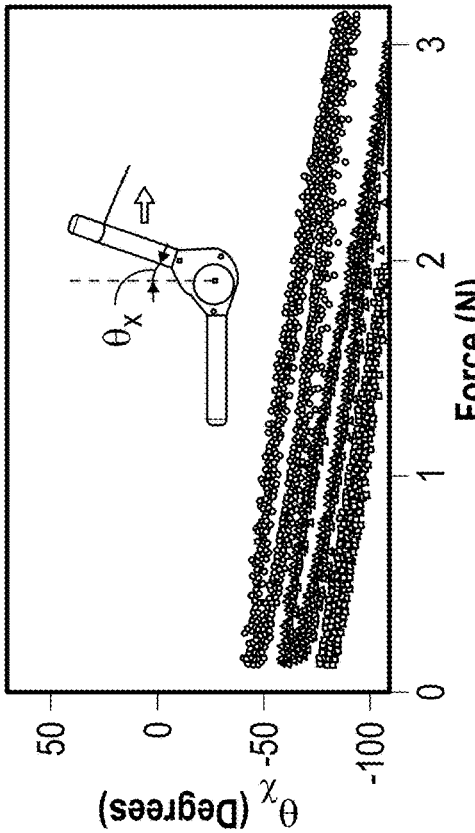


FIG. 13

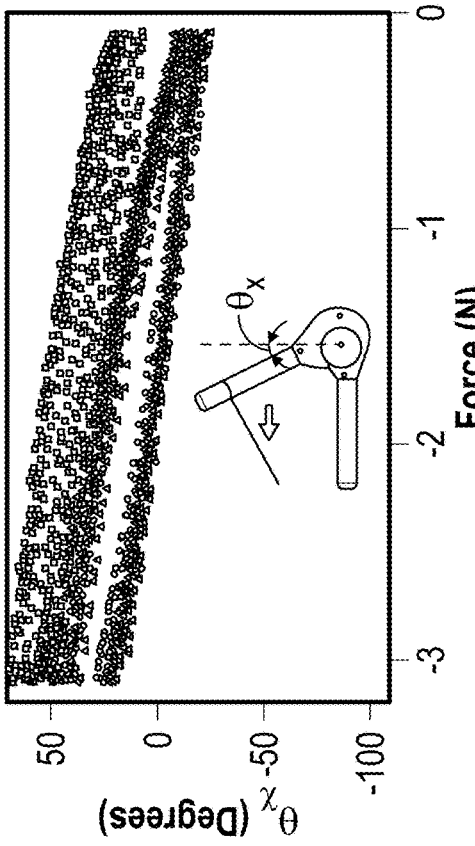


FIG. 12

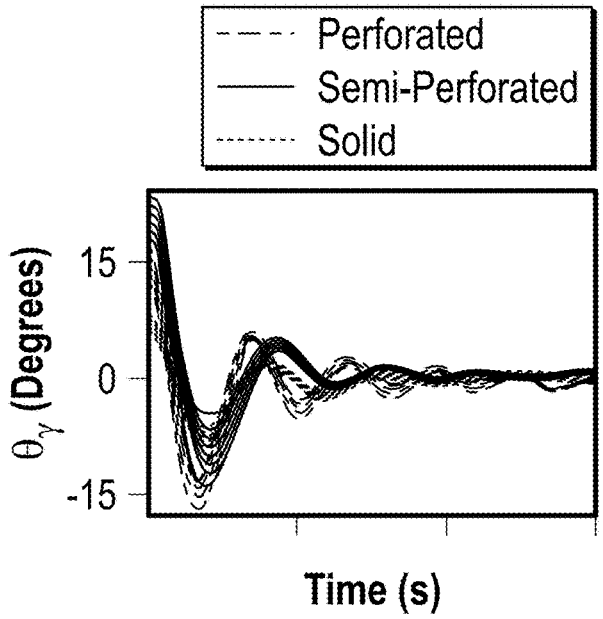


FIG. 14

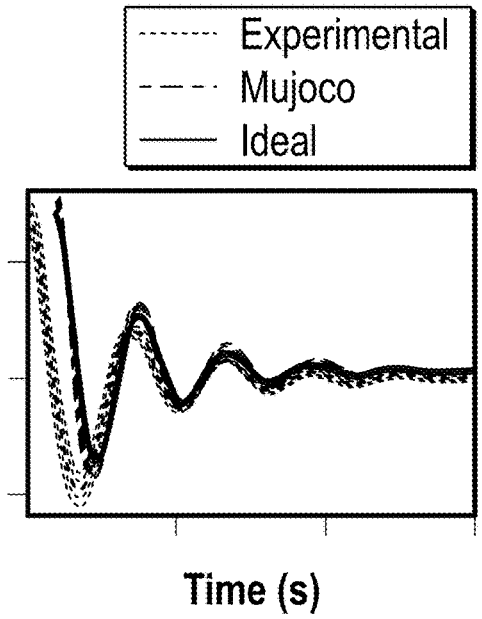


FIG. 15

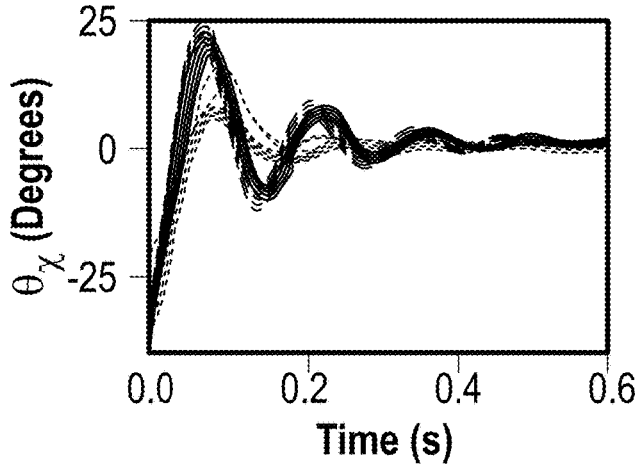


FIG. 16

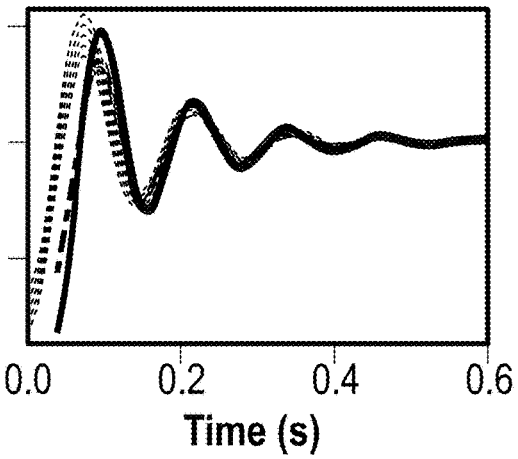


FIG. 17

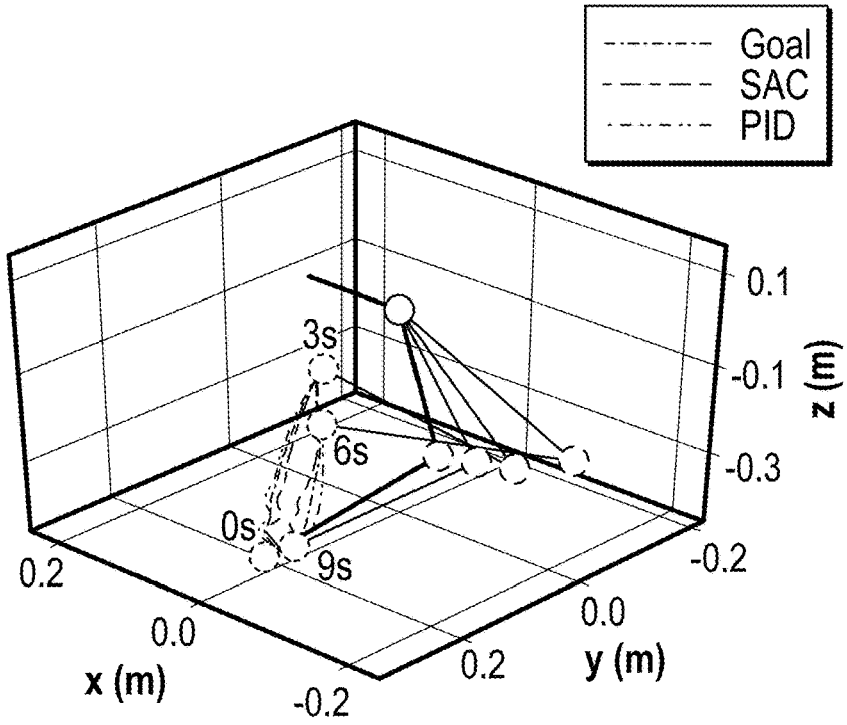


FIG. 18

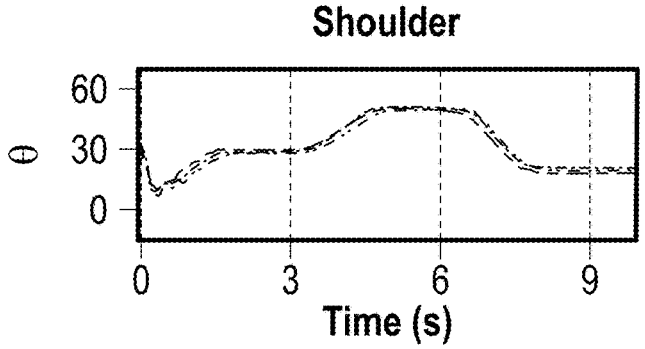


FIG. 19

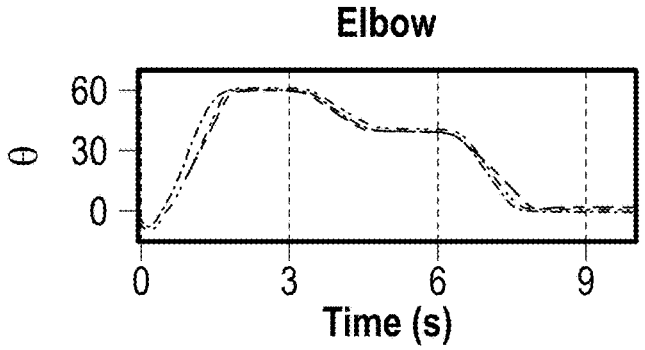


FIG. 20

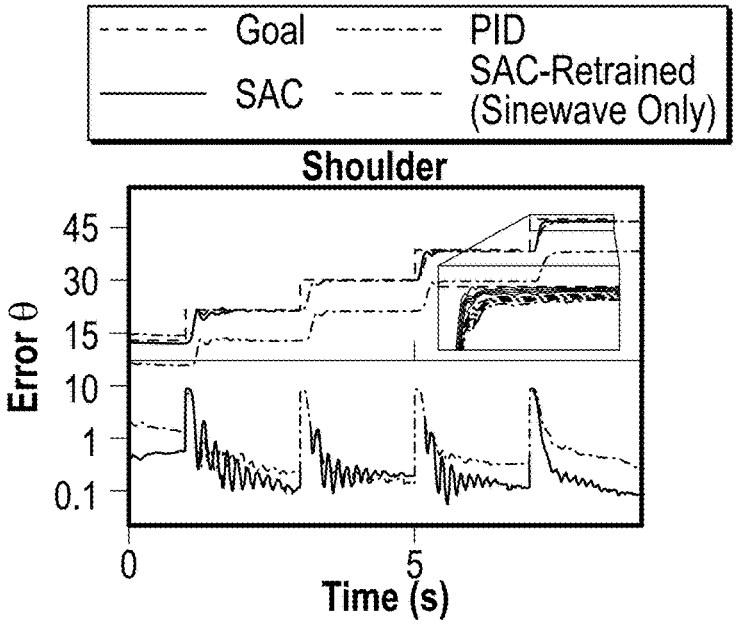


FIG. 21

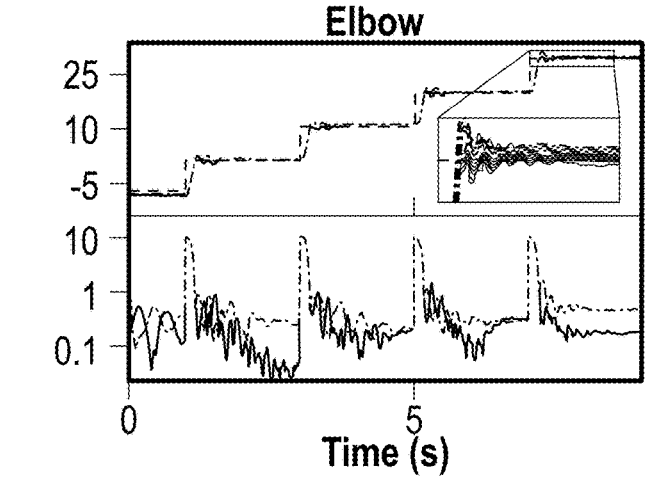


FIG. 22

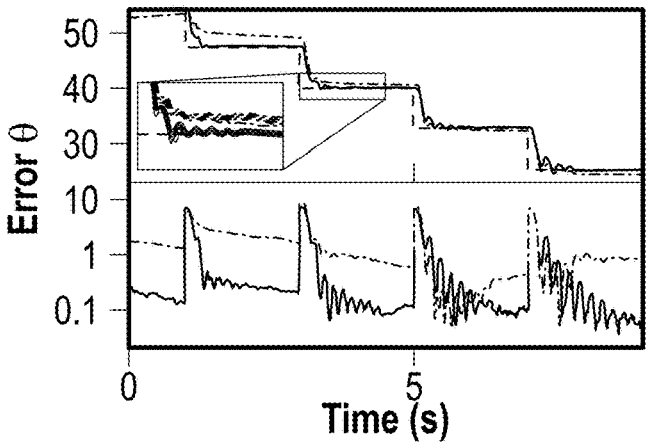


FIG. 23

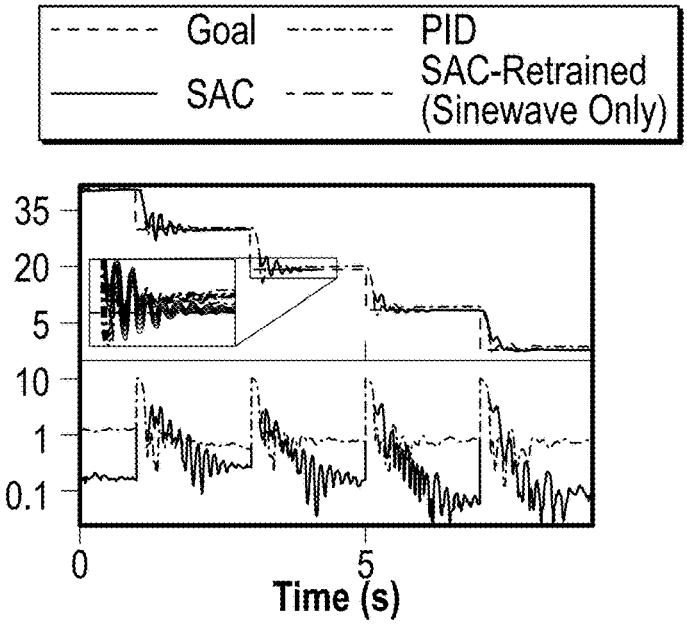


FIG. 24

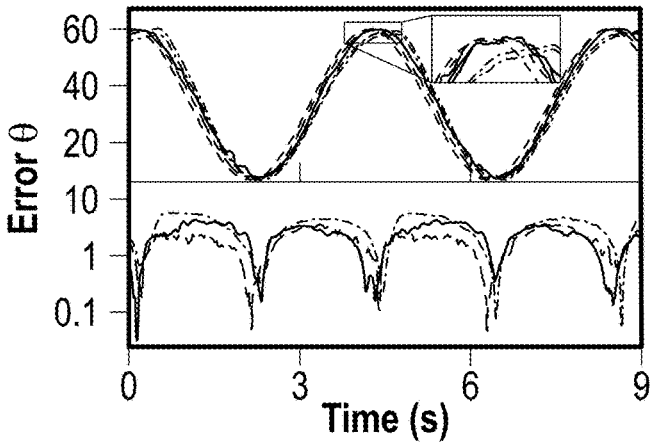


FIG. 25

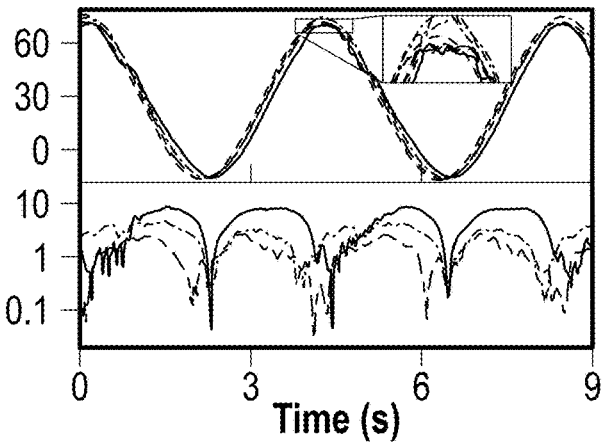


FIG. 26

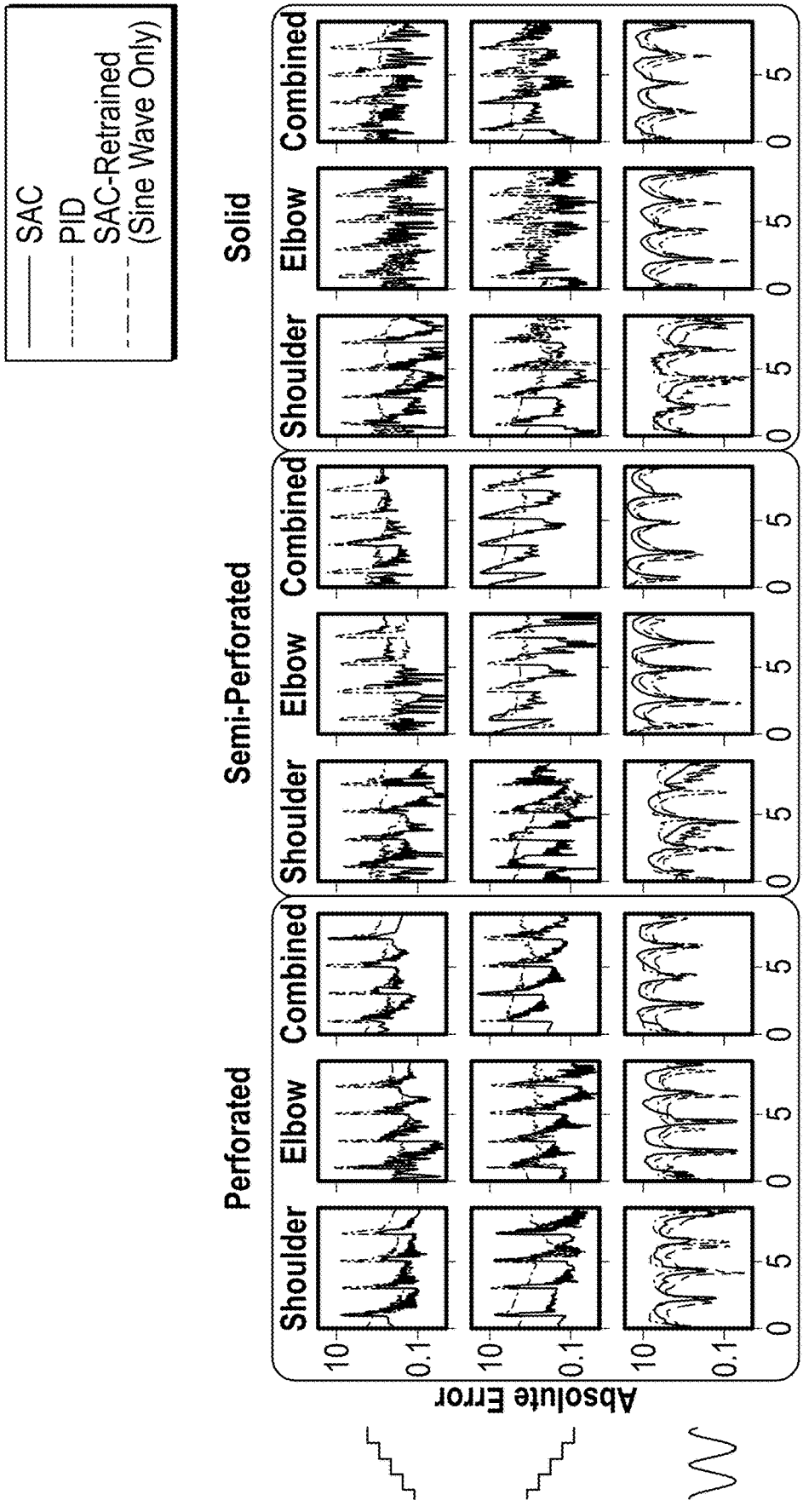

SAC
PID
SAC-Retrained
(Sine Wave Only)
Perforated
Semi-Perforated
Solid
Shoulder
Elbow
Combined
Absolute Error
10
0.1
0
5
Time (s)

FIG. 27

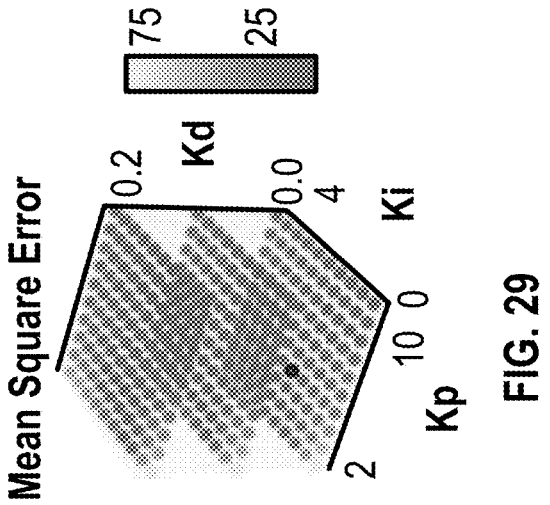


FIG. 29

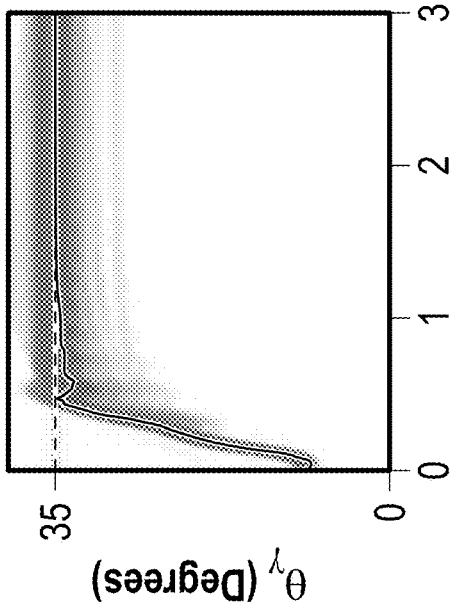


FIG. 28

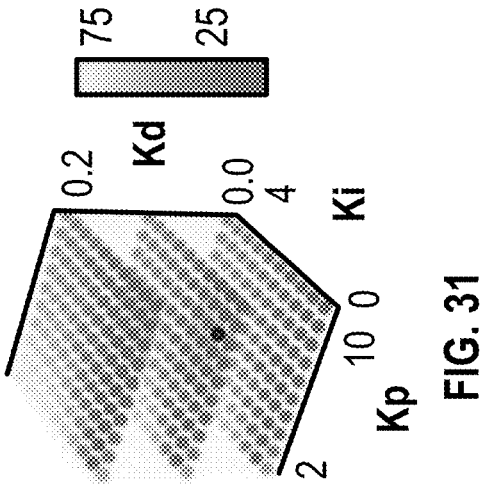


FIG. 31

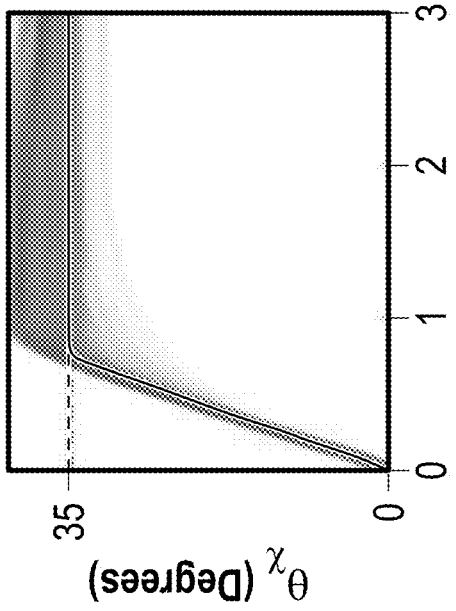


FIG. 30

SOFT-RIGID ROBOTIC JOINTS CONTROLLED BY DEEP REINFORCEMENT-LEARNING

CROSS-REFERENCE TO RELATED APPLICATION

[0001] The present application claims the benefit of and priority to U.S. Provisional Application No. 63/155,490, filed on Mar. 2, 2021, and U.S. Provisional Application No. 63/217,854, filed on Jul. 2, 2021. The entire disclosures of each of the foregoing applications are incorporated by reference herein.

BACKGROUND

[0002] The combination of elasticity and rigidity found within mammalian limbs enables dexterous manipulation, agile, and versatile behavior, yet most modern robots are either primarily soft or rigid. Most mammals have ligaments that connect bone to bone, enabling joints to passively redirect forces and softly constrain the range of motion. Hybrid robots, composed of both soft and rigid parts, promote compliance to external forces while maintaining strength and stability provided by rigid robots.

[0003] Natural manipulators, such as the human arm, have been shaped by the long-term optimization of evolution. They tend to be extremely versatile, having the dexterity to work with various objects and environments. The hybrid composition of rigid and soft components—including bones, muscles, and connective tissues—yield inherent compliance and flexibility. Biological joints often have passive stability and elasticity that create mechanical feedback that benefits disturbance responses. In addition, recent progress in the mechanical complexity of robots has popularized embedding intelligence within the system. Such robots may inherently dampen motion through elastic components or enable complicated movements emerging from simple actuation, e.g., origami robots.

[0004] In contrast, traditional robot arms tend to feature rigid components that are susceptible to large moments propagating throughout the entire robot. This means the robot's structural integrity can be compromised by a large unpredictable disturbance. Robotic manipulators involving rigid joints have strictly defined degrees of freedom resulting from the mechanical design. These joints typically fit within three categories: prismatic joints (linear movement on an axis), revolute joints (rotational movement around an axis), or a combination of the two.

[0005] Rigid robotic joints are often actuated by motors that change their position directly, allowing straightforward kinematic models to calculate the joint's position. Due to their dynamics, traditional feedback control systems, such as a proportional integral derivative (PID) controller, can solve this problem relatively well, with modifications that can adapt to the influences of gravity. One modification to account for nonlinearities involves feed-forward neural networks with PID input features.

[0006] While these dynamics are effective within controlled environments, rigid robots pose dangers to both themselves and humans because of their intrinsic inability to deal with external forces. In environments where humans are directly interfacing with robots—such as industrial manufacturing or telepresence—the robot's lack of compliance can put workers and civilians at risk of injury. Measures have been taken to increase the safety of these robots, but they are not innately safe. Factors including intrinsic safety, human detection, and control techniques influence the overall safety. The risk that a robot will cause physical harm has also been shown to moderate people's willingness to work with the robot. Strategies such as safety fences and human-detection increase safety but limit the human-robot collaboration.

[0007] Flexible robots can mitigate these external forces through structural compliance while maintaining morphological similarities with rigid robots. Systems such as soft robots and tensegrity-based robots with elastic components are inherently compliant. Biologically inspired approaches tend to exemplify this behavior. The motion of legged tensegrity structures has been validated by biological simulations while simplifying the underlying bone-ligament architecture. From a bio-mimicry perspective, a human finger has been functionally recreated through one-shot three-dimensional (3D) printing techniques employing both rigid and elastic components. Soft robots can provide safe human interaction, resulting in safer environments. Soft cable-driven exo-suits can be compliant while avoiding obstruction to the user's range of motion. Intelligent design approaches have even resulted in programmable tensegrities. However, due to the non-linearities within the elastic components, these compliant robots tend to require complex models in order to be controlled properly.

[0008] Soft robotics made from elastic components increase compliance while often sacrificing stability and precision. An accurate model of the system would enable the use of modern control techniques which can provide optimal solutions. Optimal control finds the proper control values which optimize an objective function based on the system model. A Linear-Quadratic Regulator (LQR) solves the problem of minimizing a quadratic cost matrix (encoding weights of errors, energy use, etc.) over a specified time horizon, however it is expensive and demands accurate models. Model predictive control optimizes a finite time-horizon window that is repeatedly solved at each new time-step, reducing computational cost while enabling anticipation of future events. To create the model, a common method involves system identification, which can estimate the dynamics based on measurements.

[0009] However, noise in the design process can breed inconsistencies in production, and the non-linear nature of flexible robots further complicates modeling. This emphasizes the need for control methods that can learn from data. One potential solution for controlling this variation in robots involves having precisely adjusted models for each physical instance. But these approaches are cumbersome due to the requirement of constructing precise models. Thus, there is a need for a system and method of controlling soft-rigid hybrid robotic joints that overcome the deficiencies of the conventional control methodologies.

SUMMARY

[0010] The present disclosure provides a soft-rigid hybrid robotic joint with variable kinetic parameters that are controllable through learned methods. In particular, the present disclosure provides a novel approach to designing and constructing a hybrid joint through parametric design choices that adjust dynamic properties of the system. Deal-

ing with the inherent modularity and variability necessitates a model-free controller that adjusts to new contexts in a relatively short time.

[0011] Reinforcement learning (RL) solves the above-noted deficiencies of conventional control methods through a data driven approach to solve the optimal control problem. Deep-reinforcement learning (DRL) extends this framework by using neural networks as parameter estimators. In DRL, algorithms solve for an optimal policy that will compute the control value (or action) from the current state of the system. As used herein, the term "model-free" denotes control methods that do not explicitly model the dynamics of the system.

[0012] DRL solves control problems through gathering data, surpassing human skill in the domain of games, teaching legged robots to walk, and much more. However, neural algorithms are data intensive and gathering data on robots can be incredibly costly in machine-hours, making sample efficiency important. Transfer learning provides a method for reducing training time by using prior knowledge. Whether through examples of expert trajectories or an agent trained in simulation, transfer learning involves an agent leveraging training experience from some domain A to perform better on a new task in a similar domain B. In the case of robotics, simulations provide a route to shortening real world training times through pre-trained models that can be fine-tuned on the physical robot.

[0013] The present disclosure provides a methodology for the construction of novel robotic joints with a hybrid soft-rigid structure and model-free control strategies for this class of joints. First, the joint design begins with a simplistic 3D printing approach enabling easy customization with a molding process that supports interfacing between soft and rigid parts. Customization includes modifying the perforation of the rigid components, resulting in three types of joints (i.e., perforated, semi-perforated, and solid), each with distinct ligament meshes.

[0014] Second, the joints c characterized through spring and damper experiments to provide insight in how the modification changes the dynamics. These experiments expose significant nonlinearities in the system and identify how design parameters can modulate the spring-damper terms, leading to future work of programmable dynamics. The parameters found inform a simulation, e.g., MuJoCo, enabling the use of transfer learning, e.g., transferring machine learned knowledge/algorithm from a simulation to a physical/real world. As used herein, "transfer learning" denotes training of a control algorithm that includes a DRL algorithm on a simulation (e.g., of the joint) and transferring the algorithm resulting from the training to control of the physical object, namely, the robotic joint.

[0015] The present disclosure also describes and evaluates two model-free control methodologies which require minimal user input: a proportional-integral-derivative (PID) controller and a Soft Actor-Critic (SAC) algorithm. Three tasks were designed to evaluate the accuracy of each controller. The tasks were termed: 1) increasing steps, 2) decreasing steps, and 3) sinewave. The SAC policy yields greater precision through a wider range than the tuned PID controller, and robustly solves the presented tasks by generalizing through the disclosed joint designs. Thus, the present disclosure provides a method for designing soft-rigid robotic joints with parameterizable stiffness and damping. Model-free DRL is used to compensate for variability and non-linearities.

[0016] According to one embodiment of the present disclosure, a robotic arm is disclosed. The robotic arm includes a first link, a second link, and a joint interconnecting the first link and the second link, such that the first link is movable relative to the second link along an axis of motion. The joint includes: a socket component coupled to a distal end portion of the second link and a ball component coupled to a proximal end portion of first link, the ball component is configured to rotationally fit within the socket component. The joint also includes a flexible membrane encasing the socket component and the ball component.

[0017] Implementations of the above embodiments may include the following modifications. According to one aspect of the above embodiment, the socket component may include a first plurality of through-holes. The ball component may include a second plurality of through-holes. The first plurality of through-holes may be disposed in a first socket plane and a second socket plane. The first socket plane and the second socket plane may be perpendicular to each other and are aligned with the axis of motion. The second plurality of through-holes may be disposed in a first ball plane and a second ball plane. The first ball plane and the second ball plane may be perpendicular to each other and are aligned with the axis of motion.

[0018] According to another aspect of the above embodiment, the flexible membrane may be disposed within a space defined by the first plurality of through-holes and the second plurality of through-holes. The flexible membrane may be formed from an elastomer.

[0019] According to a further aspect of the above embodiment, the robotic arm may also include an actuator coupled to the second link and a cable coupled to the first link, wherein the actuator is configured to move the first link by spooling the cable. Each of the socket component and the ball component may include at least one routing block configured to route the cable around the joint.

[0020] According to another embodiment of the present disclosure, a method of manufacturing a robotic joint is disclosed. The method includes forming a first plurality of through-holes in a ball component; forming a second plurality of through-holes in a socket component; and inserting the ball component into the socket component to form a robotic joint, wherein the ball component is movable relative to the socket component along an axis of motion; and applying a flexible membrane over the robotic joint and within a space defined by the first plurality of through-holes and the second plurality of through-holes.

[0021] Implementations of the above embodiments may include the following modifications. According to one aspect of the above embodiment, applying the flexible membrane includes placing the robotic joint in a mold and pouring a liquid precursor composition of the flexible membrane.

[0022] According to another aspect of the above embodiment, the first plurality of through-holes may be disposed in a first socket plane and a second socket plane and the first socket plane and the second socket plane may be perpendicular to each other and are aligned with the axis of motion. The second plurality of through-holes may be disposed in a first ball plane and a second ball plane and the first ball plane

and the second ball plane may be perpendicular to each other and are aligned with the axis of motion.

[0023] According to another embodiment of the present disclosure, a method for programming a control agent for controlling a robotic arm is disclosed. The method includes running a simulation of a robotic arm based on a plurality of physical parameters on a workstation; training a reinforcement learning algorithm based on the simulation of the robotic arm; and loading a control agent based on the reinforcement learning algorithm into a controller controlling the robotic arm.

[0024] Implementations of the above embodiments may include the following modifications. According to one aspect of the above embodiment, training may further include performing a plurality of tasks to reach a random point a three-dimensional space of the simulation. Training may also include implementing a reward function configured to minimize distance traveled to the random point. According to another aspect of the above embodiment, the method may also include retraining the reinforcement learning algorithm based on operation of the robotic arm in a physical space.

BRIEF DESCRIPTION OF DRAWINGS

[0025] Various embodiments of the present disclosure are described herein below with reference to the figures wherein:

[0026] FIG. **1** is a perspective view of a robotic arm having a hybrid soft-rigid shoulder and elbow joints according to the present disclosure;

[0027] FIG. **2** is a perspective view of a 3D model of the robotic arm of FIG. **1** according to the present disclosure;

[0028] FIG. **3** is a side view of 3D model of the shoulder joint according to the present disclosure;

[0029] FIG. **4** is a side view of 3D model of the elbow joint according to the present disclosure;

[0030] FIG. **5** is a partial, cross-sectional view of the shoulder joint according to the present disclosure;

[0031] FIG. **6** is a partial, cross-sectional view of the elbow joint according to the present disclosure;

[0032] FIG. **7** is a side view of a perforated ball component of the shoulder joint according to one embodiment of the present disclosure;

[0033] FIG. **8** is a side view of a semi-perforated perforated ball component of the shoulder joint according to another embodiment of the present disclosure;

[0034] FIG. **9** is a side view of a solid ball component of the shoulder joint according to another embodiment of the present disclosure;

[0035] FIG. **10** is a plot of force measured as a function of an angle during closing of the shoulder joint according to the present disclosure;

[0036] FIG. **11** is a plot of force measured as a function of an angle during opening of the shoulder joint according to the present disclosure;

[0037] FIG. **12** is a plot of force measured as a function of an angle during closing of the elbow joint according to the present disclosure;

[0038] FIG. **13** is a plot of force measured as a function of an angle during opening of the elbow joint according to the present disclosure;

[0039] FIG. **14** are plots (angle as a function of time) of drop test performed on robotic arms having a shoulder joint with a ball component being one of the three types (perforated, semi-perforated, solid) according to the present disclosure;

[0040] FIG. **15** are plots (angle as a function of time) of a drop test performed on a robotic arm having a shoulder joint with a perforated ball component, a simulated drop test, and an idealized drop test, according to the present disclosure;

[0041] FIG. **16** are plots (angle as a function of time) of drop test performed on robotic arms having an elbow joint with a ball component being one of the three types (perforated, semi-perforated, solid) according to the present disclosure;

[0042] FIG. **17** are plots (angle as a function of time) of a drop test performed on a robotic arm having an elbow joint with a perforated ball component, a simulated drop test, and an idealized drop test, according to the present disclosure;

[0043] FIG. **18** shows a 3D visualization of an end effector trajectory moved by a robotic arm according to the present disclosure;

[0044] FIG. **19** shows a plot of an angle of the shoulder joint as a function of time of the robotic arm moving the end effector of FIG. **18**;

[0045] FIG. **20** shows a plot of an angle of the elbow joint as a function of time of the robotic arm moving the end effector of FIG. **18**;

[0046] FIG. **21** shows plots of a measured angle of the shoulder joint while performing an increasing steps maneuver using a SAC algorithm and a PID algorithm and the resulting error for each;

[0047] FIG. **22** shows plots of a measured angle of the elbow joint while performing an increasing steps maneuver using a SAC algorithm and a PID algorithm and the resulting error for each;

[0048] FIG. **23** shows plots of a measured angle of the shoulder joint while performing a decreasing steps maneuver using a SAC algorithm and a PID algorithm and the resulting error for each;

[0049] FIG. **24** shows plots of a measured angle of the elbow joint while performing a decreasing steps maneuver using a SAC algorithm and a PID algorithm and the resulting error for each;

[0050] FIGS. **25** and **26** show goal (ideal) plots of a measured angle of the shoulder joint while performing a sine wave maneuver along with plots using a SAC algorithm, a PID algorithm, a retrained SAC algorithm and the resulting error for each;

[0051] FIG. **27** shows absolute error plots as a function of time for each of three types of joint components, for each of the shoulder joint, the elbow joint, and the robotic arm, performing three motions (increasing steps, decreasing steps, and sinus wave);

[0052] FIG. **28** shows a tuning plot for reaching an angle of about 35° using a PID control algorithm for the shoulder joint;

[0053] FIG. **29** shows a 3D visualization grid search values for the shoulder joint for each of the K_P, K_1, and K_D parameters of the PID control algorithm according to the present disclosure;

[0054] FIG. **30** shows a tuning plot for reaching an angle of about 35° using a PID control algorithm for the elbow joint; and

[0055] FIG. **31** shows a 3D visualization grid search values for the elbow joint for each of the K_P, K_1, and K_D parameters of the PID control algorithm according to the present disclosure.

DETAILED DESCRIPTION

[0056] Embodiments of the presently disclosed robotic system including a robotic arm are described in detail with reference to the drawings, in which like reference numerals designate identical or corresponding elements in each of the several views. As used herein the term "distal" refers to the portion of the robotic arm that is closer to an end effector held by the robotic arm, while the term "proximal" refers to the portion that is farther from the end effector.

[0057] The terms "application" and/or "software" are used interchangeably and may include a computer program designed to perform functions, tasks, or activities for the benefit of a user. Application may refer to, for example, software running locally or remotely, as a standalone program or in a web browser, or other software which would be understood by one skilled in the art to be an application. An application may run on a controller, or on a user device, including, for example, a mobile device, a personal computer, or a server system.

[0058] The present disclosure provides a robotic arm having one or more hybrid soft-rigid joints. With reference to FIG. **1**, a robotic arm **10** includes a first link **12**, a second link **14**, and a third link **16**. The links **12**, **14**, **16** may be formed from any suitable rigid material, such as metal, carbon fiber, polymer, etc. The links **12**, **14**, **16** may have any suitable shape such as rods, bars, hollow housings, and etc. Each of the links **12**, **14**, **16** includes a distal end portion **12***a*, **14***a*, **16***a*, and a proximal end portion **12***b*, **14***b*, **16***b*, respectively. The first link **12** may include an end effector **13** (e.g., grasper) coupled to the distal end portion **12***a*. The first link **12** may also include a sensor **15**, which may be disposed on the distal end portion **12***a* or on any other part of the robotic arm **10**. In embodiments, multiple sensors **15** may be used to measure operational parameters of the robotic arm **10**. The sensor **15** may be an inertial measurement unit (IMU) configured to measure angular motion, roll, pitch, yaw, and other motion parameters. The sensor **15** may include accelerometers, gyroscopes, magnetometer, and other sensors suitable for measuring these motion parameters.

[0059] The first and second links **12** and **14** are interconnected by a first joint **20**, which acts like an elbow joint and the second and third links **14** and **16** are interconnected by a second joint **30**, which acts like a shoulder joint. Each of the first and second joints **20** and **30** are coupled to a first actuator **40** and a second actuator **50**, respectively. Each of the first and second actuators **40** and **50** may be motors coupled to first and second spools **42** and **52**, respectively. The first and second actuators **40** and **50** may include various sensors, such as torque sensors, angular sensors, encoders, and the like, configured to measure force imparted by the first and second actuators **40** and **50** and angle between the first, second, and third links **12**, **14**, **16**.

[0060] The first actuator **40** and the first spool **42** are coupled to the second link **14**, namely, the distal end portion **14***a*. The first spool **42** includes a cable **44** securely coupled to the first link **12**, such that the first spool **42** is coupled proximally with the first joint **20** and the cable **44** is coupled distally of the first joint **20**, i.e., the proximal portion **12***b* of the first link **12**. Thus, as the first actuator **40** spools and unspools the cable **44**, the first link **12** moves relative to the second link **14** about the first joint **20** and in particular, as the cable **44** is spooled, a first (i.e., elbow) angle between the first and second links **12** and **14** decreases and conversely, as the cable **44** is unspooled, the first angle increases.

[0061] The second actuator **50** and the second spool **52** are coupled to the third link **16**, i.e., distal end portion **16***b*. The second spool **52** includes a cable **54** securely coupled to the second link **14**, proximal end portion **14***b*, such that the second spool **52** is coupled proximally of the second joint **30** and the cable **54** is coupled distally of the second joint **30**. The cable **54** is routed through routing blocks **55***a*, **55***b*, **55***c*. The routing block **55***a* is disposed on the proximal end portion **16** a of the third link **16** and over a socket component **34** of the second joint **30** (FIG. **3**). The routing blocks **55***b* and **55***c* are disposed on a distal end portion of the second link **14** and over a ball component **32** of the second joint **30**. In embodiments, any suitable number of routing blocks or configurations may be used to minimize the forces and angle of attack of the cables relative to the links of the robotic arm.

[0062] During operation, as the second actuator **50** spools and unspools the cable **54**, the second link **14** moves relative to the third link **16** about the second joint **30** and in particular, as the cable is spooled, a second (i.e., shoulder) angle between the first and second links **12** and **14** decreases and conversely, as the cable **54** is unspooled, the second angle increases. In addition to a cable and spool system, various other mechanical linkages may be used, such as a belt, a drive rod, and the like.

[0063] With reference to FIGS. **2-6**, each of the first and second joints **20** and **30** are ball and socket joints. The first joint **20** includes a ball component **22** and a socket component **24**. Similarly, the second joint **30** includes a ball component **32** and a socket component **34**. The ball components **22** and **32** as well as the socket components **24** and **34** may be solid or hollow and may have a plurality of through-holes **35** formed therethrough. The ball components **22** and **32** may be formed from any suitable rigid material, such as polymers (e.g., polylactic acid), metals, ceramics, and combinations thereof. The ball components **22** and **32** may be formed using subtractive or additive manufacturing (e.g., machining, laser cutting, 3D printing, etc.)

[0064] With reference to FIGS. **7-9**, various embodiments of the components are shown with respect to the ball component **32**. Same modifications may be made to other components and only the ball component **32** is described for the sake of brevity. As shown in FIG. **7**, the ball component **32** includes through-holes **35** disposed in a square grid pattern such that the through-holes **35** and a distance therebetween covers the majority of the surface of the ball component **32**. This pattern is denoted herein "perforated". As used herein, "through-holes" denotes a pair of holes that lie on the same longitudinal axis. In embodiments, where the joint components are solid, the through-holes may define a lumen through the component (i.e., ball components **22** and **32** and socket components **24** and **34**).

[0065] In embodiments, the through-holes **35** may be disposed in any suitable pattern, such as rectangular, circular (concentric circles), and the like. The through-holes **35** may be disposed in a plurality of planes. As shown in FIG. **7**, the through-holes **35** lie in two perpendicular planes, i.e., x and y planes, which are aligned with the axis of intended motion.

In particular, the primary axes of rotation for the second joint **30** is the θy direction (FIG. **3**), and the first joint **20** as the θx direction (FIG. **4**).

[0066] With reference to FIG. **8**, the ball component **32** includes through-holes **35** that are disposed in a "semi-perforated" pattern since there are fewer through-holes **35** than in the "perforated" pattern of FIG. **8**. The through-holes **35** are disposed along diagonal axes "a-a" and "b-b" which lie on the x plane. Similarly, the through-holes **35** disposed on the y plane are disposed along corresponding diagonal axes (not shown.) FIG. **9** shows a solid ball component **32** that is devoid of any through holes. As noted above, each of the components may be hollow, including the solid ball component **32** of FIG. **9**.

[0067] To form the joints **20** and **30** the ball components are cast in a flexible membrane **37**, which is disposed over the ball components **22** and **32** and socket components **24** and **34**. The membrane **37** may be formed from any suitable compliant, flexible, and elastic polymer, i.e., elastomers. Suitable elastomers include rubbers, such as natural rubbers, silicone rubbers ethylene propylene rubber, ethylene propylene diene rubber, and nitrile rubbers and synthetic elastomers such as styrene-butadienes, polyisoprenes, polybutadienes, polysiloxanes, fluoroelastomers, polyurethane elastomers, and the like. The ball components **22** and **32** and socket components **24** and **34** are joined together, such that the ball components **22** and **32** rotationally fits within their counterpart socket components **24** and **34** and the membrane **37** is applied thereto. The ball components **22** and **32** fit within their counterpart socket components **24** and **34**, which are then placed within mold supports **39** (FIGS. **5** and **6**) and the liquid precursor composition of the membrane **37** is poured into the mold supports **39** to form the membrane **37** after curing, which may be from about **15** minutes to about **5** hours, depending on the curing times of the polymer. In addition, the liquid composition penetrates the ball components **22** and **32** and the socket components **24** and **34** via the through-holes **35** as well as their interconnecting lumens. This allows the membrane **37** to encase the joints **20** and **30** and acts as connective tissue within the components of the joints **20** and **30**. In further embodiments, the membrane **37** may be formed by casting, dipping, layering, calendaring, spraying, and combinations thereof

[0068] Adjusting degree of perforation (i.e., number of through-holes **35** and lumens therebetween) the ball components **22** and **32** and socket components **24** and **34** changes the amount of negative (i.e., empty) space, leading to changes in the amount of the membrane **37** that penetrates the ball components **22** and **32** and socket components **24** and **34**, which in turn, modulates dynamics of the joints **20** and **30**. In particular, the combination of the membrane **37** surrounding the joints **20** and **30** and within the through-holes **35** creates a system of spring-dampers which can be modified by design parameters including hole quantity, outer-membrane width, and choice of flexible material used in forming the membrane **37**. Thus, the joint dynamics may be adjusted by varying the quantity of through-holes **35** in the elastic membrane **37** of each joint. The through-holes **35** act as parallel spring-damper systems, effectively adjusting the non-linear dynamics.

[0069] The robotic arm **10** may be controlled using any suitable control methodologies including model-free methodologies such as a proportional-integral-derivative (PID) controller and the model-free reinforcement learning algorithm, such as soft actor critic (SAC) algorithm. The robotic arm **10** may also be controlled using model-based control techniques.

[0070] In order to generate a suitable control agent for controlling movement of the robotic arm **10**, the robotic arm **10** may be modeled using spring-damper characterization, e.g., through force-deflection experiments as described in the "Examples" section. The sensors **15** as well as the sensors of the actuators **40** and **50** may be used to measure various parameters (e.g., force). These parameters along with dimensions of the robotic arm **10** may be used to generate a simulation of the robotic arm **10** (e.g., a MuJoCo simulator as described in the "Examples" section). The simulation of the robotic arm **10** may be used to train a control agent, which in embodiments may be machine learning algorithm as described in Example **4** below. Suitable machine learning algorithms include model-free reinforcement learning techniques such as Proximal Policy Optimization (PPO), Trust Region Policy Optimization (TRPO), Asynchronous Advantaged Actor Critic (A3C) and Soft Actor Critic (SAC).

[0071] The terms "artificial intelligence," "data models," or "machine learning" may include, but are not limited to, neural networks, convolutional neural networks (CNN), recurrent neural networks (RNN), generative adversarial networks (GAN), Bayesian Regression, Naive Bayes, nearest neighbors, least squares, means, and support vector regression, among other data science and artificial science techniques.

[0072] In various embodiments, the neural network may include a temporal convolutional network, with one or more fully connected layers, or a feed forward network. Training of the neural network may happen on a separate system, e.g., workstations with graphic processor unit ("GPU"), high performing computer clusters, etc., and the trained algorithm would then be deployed to a computer **100** (FIG. **1**) controlling the robotic arm **10**. The computer **100** may include any suitable processor (not shown) operably connected to a memory (not shown), which may include one or more of volatile, non-volatile, magnetic, optical, or electrical media, such as read-only memory (ROM), random access memory (RAM), electrically-erasable programmable ROM (EEPROM), non-volatile RAM (NVRAM), or flash memory. The processor may be any suitable processor (e.g., control circuit) adapted to perform the operations, calculations, and/or set of instructions described in the present disclosure including, but not limited to, a hardware processor, a field programmable gate array (FPGA), a digital signal processor (DSP), a central processing unit (CPU), a microprocessor, and combinations thereof. Those skilled in the art will appreciate that the processor may be substituted for by using any logic processor (e.g., control circuit) adapted to execute algorithms, calculations, and/or set of instructions described herein.

[0073] The following Examples illustrate embodiments of the present disclosure. These Examples are intended to be illustrative only and are not intended to limit the scope of the present disclosure.

EXAMPLE 1

[0074] This Example describes construction of an exemplary robotic arm according to the present disclosure.

[0075] Carbon fiber rods with a 13 mm outer diameter were used as artificial bones linking the joints as shown in

FIG. **1**. EcoFlex 00-50 (Smooth-On), was used as the passive elastic component due to its ability to deform, and its usage in fabricating soft robots (53-55) having tensile strength of 315 psi, 12 psi 100% modulus, and 980% elongation at break.

[0076] Polylactic acid (PLA) was used for the rigid components of the joints, motor-mounts, sensor-mounts, and cable routing. Three types of joints were prepared—perforated, semi-perforated, and solid. The computer-aided design (CAD) modeled joints and mold designs were printed with a Prusa i3 MK3 3D printer. Once inserted on the mold supports as seen in FIGS. **5** and **6**, the mold was sealed with an easily removable adhesive such as hot glue. After mixing a 1:1 ratio of Ecoflex-50, the silicone material was poured into the mold and is left to cure for approximately 3 hours. The final product is shown in FIG. **1**: a hybrid joint containing rigid components housed within a soft silicone mesh.

[0077] BNO55 IMU sensors were used to measure operating parameters of the robotic arm. The IC BNO55 IMU sensor includes a built-in sensor fusion that allows for a 100 Hz sampling rate, returning the axis angles: roll, pitch, and yaw. 12V DC motors with 4.2 kg/cm were used as actuators. A custom PCB was designed to interface with the electronics and communicate with a desktop computer using I2C serial communication. In order to simulate and process large amounts of data, a PC with an Intel Core i7-9700 K processor was used to speed up training time.

[0078] The motor mounts were attached to the carbon fiber rods at roughly the location of the major muscles of a human arm. The cables acting on the carbon fiber rods transmit a force similar to tendons pulling on bones. The biceps brachialis flexes the elbow joint—increasing θx as denoted in FIG. **4**—and thus the corresponding motor mount is placed on the upper arm with the cable connecting to the lower arm. The medial deltoid is the prime mover for shoulder abduction, which is seen as positively increasing θy denoted in FIG. **3**. Accordingly, the second motor mount is located to simulate the force created by the medial deltoid. Together the placement gives the ability of shoulder abduction θy and biceps flexion θx.

[0079] When directly wrapping the cables around the shoulder joint, the forces required to cause movement were too large for the chosen motors due to the angle of attack—which is created by the cable attaching the shoulder's motor and the upper-arm—is too small to transmit the necessary forces. Another issue stemming from the cable wrapping around the joint was that it can lose its pathing, falling to either side of the joint rather than tracing the desired route. Routing blocks were used to route the cable for the shoulder joint as shown in FIG. **3**. Bearings were attached for less cable friction. Within Autodesk Inventor, the measured angle of attack before was θa=59.1° and after adding the cable routing was measured as θb=22.5°. Thus Ft=Fmotdcos (θ) calculates the transfer force Ft from the motor force Fm. The chosen cable routing offers about 1.8 times more force.

EXAMPLE 2

[0080] This Example describes simulation modeling and evaluation of the exemplary robotic arms of Example 1.

[0081] The setup of the spring experiment involved anchoring each joint with the proximal portion meeting a rigid testbed and the distal end attaching to a single rod with an IMU. Force was measured through a calibrated load cell, which records simultaneously alongside the IMU. Forces were applied perpendicular to the joint angle, as depicted in FIGS. **10-13**, which show opening and closing plots for both joints, in a quasistatic manner. More than **10** repeated trials were run for both directions of each of the shoulder and elbow joints. The starting angles were determined by the resting point after the first trial which was removed from the data.

[0082] The torsional spring coefficients were characterized through force-deflection experiments shown in plots of FIGS. **10-13**. Each joint was anchored to the testbed by the proximal end of the link, and an IMU was attached distally. The experiments of joint opening seen in FIGS. **10** and **12** has the data circling counterclockwise while joint closing seen in FIGS. **11** and **13** circles clockwise in time. The data shows that increasing the number of holes—which increases the aggregate cross-sectional surface area of the rubber—requires more force to get to the same angular displacement during joint opening. The results are less clear during joint closing. The force required to close the joint depends on both the volume of silicone material within the through-holes and external membrane thickness surrounding the rigid components. In the plots of FIGS. **10-13** the number of through-holes in each of the perforated, semi-perforated, and solid joints are easily distinguishable as a function of the angle and force used to open/close the joint. The graphs appear discontinuous at 0 N due to mechanical hysteresis and slipping within the joint. Hysteresis in FIGS. **10-13** represents graphically as a belly and demonstrates non-linearities; different forces are required depending on the directionality of joint movement. This characterization supports our need for data-driven non-linear estimators.

[0083] Damping experiments were conducted using a drop-test. The resulting oscillations are shown in FIGS. **14** and **16**, which were characterized by fitting an exponential decay equation to the positive peaks of the data. These experiments provide more evidence that varying the design allows for control over dynamic parameters.

[0084] For the damping experiments the proximal side of the joint was attached to the testbed while the distal portion of the joint connected to the rest of the arm. In the case of the shoulder this includes the upper arm, lower arm, elbow joint and mounted parts. For the elbow, the experiment only included the lower arm. Each joint was displaced by around 75° and released. Data was cropped at the peak of the first oscillation to align the separate trials. The motion after the first peak is shown in FIGS. **14** and **16**. The envelopes of the peaks were used to fit the exponential decay equation to the data. This resulted in the time constant τ. The damped angular frequency ω_d was calculated from the inter-peak times resulting in the necessary parameters for modeling.

[0085] Characterization of the robotic arms dynamics provided insights and analysis which helped build more accurate models and control schemes. From these experiments and the calculations in "Modeling spring-damper", the time constant τ and the damping ratio ζ were acquired to parameterize both an ideal second order spring-damper and the MuJoCo simulator. The ideal equation was used to validate MuJoCo. Both approaches were compared to the experimental data in FIGS. **15** and **17**. The nonlinearities emerge as differing resonant frequencies between the initial high-amplitude transient and the tail. A small time shift of 4 ms was utilized to align with the tail since it is closer to the small-deviation linear dynamics. These modeling results

provide a sufficiently accurate foundation for experimenting with control strategies—specifically reinforcement learning algorithms—in simulation.

[0086] The robotic arm was simplified as a revolute joint for each corresponding actuated axis. Each joint was modeled as a torsional spring-damper system. The ideal equation for a damped harmonic oscillator is shown in formula (1) below:

$$y=ae^{-t/\tau}\cos(\omega_d t) \tag{1}$$

[0087] where ωd is the damped angular frequency and τ is the time constant. The relation between the damped angular frequency and the natural frequency of the system follows from $\omega_d=\omega_n\sqrt{1-\zeta^2}$. The natural frequency wn can be found using $\omega_n=1/(\zeta\tau)$. Finally, solving for results in formula (2):

$$\zeta = \frac{1}{\sqrt{(\tau\omega_d)^2+1}} \tag{2}$$

[0088] These equations result in the ideal characterization which is shown on the right in FIGS. **15** and **17** for shoulder and elbow joints, respectively, alongside the experimental data and MuJoCo model.

[0089] MuJoCo is a multi joint rigid body simulator specializing in robotics and biomechanics which uses different solvers to compute forward and inverse dynamics and kinematics. MuJoCo is highly optimized, having higher relative speeds for robotic-like tasks while maintaining the top precision compared to other popular simulators (other than in the case of many disjoint bodies). MuJoCo also has been used for designing environments for reinforcement learning experimentation through OpenAI Gym. Rigid bodies can be created or imported, assigned material properties, and actuated around specified joints. Internal parameters can be set to manipulate joint and actuator dynamics. In order to mirror the physical robot, the 3D models of the robotic arms of Example 1 were imported into MuJoCo. The weights of each piece were set, along with the parameters τ and ζ for the spring-damper system. The resulting graphs validating the accuracy of MuJoCo are shown in FIGS. **15** and **17**, comparing the plots of drop tests with ideal and simulated (MuJoCo) plots.

EXAMPLE 3

[0090] This Example describes PID control algorithm for controlling the exemplary robotic arms of Example 1.

[0091] The PID control algorithm was adjusted using the robotic arm with perforated joints. The control function evaluated is shown in formula (3):

$$u(t) = Kpe(t) + Ki\int_O^{\tau} e(t)dt\ Kd\frac{de(t)}{dt} \tag{3}$$

[0092] where u(t) is the control value output, KP is the proportional term, KI is the integrator term, KD is the derivative term, and e(t) is the error for the time-step t.

[0093] To tune the controller, a grid search was used to test **300** different values for the shoulder and elbow joint. MSE was calculated using formula (4):

$$MSE = \frac{1}{T}\sum_{t=0}^{T}(yt - y\hat{t})^2 \tag{4}$$

[0094] with T as the total time-steps, yt as the measured value, and yt as the goal value. MSE was used to determine the parameters resulting in the minimum error to a desired set point, formalized using formula (5):

$$(K_P, K_I, K_D) = \arg\frac{\min}{K_P, K_I, K_d} MSE(K_P, K_I, K_D) \tag{5}$$

[0095] The ranges for the grid search were chosen based off preliminary hand-tuning, and used ten K_P values, ten K_1 values and three K_D values (since the derivative term is often dropped). The results are shown in FIGS. **28-31** and Table 1. A grid search over **300** parameter combinations was used to tune the PID controller. Each trial involved reaching the set point of 35° which resides near the center of the workspace. FIGS. **28** and **30** show each trial of the grid, with a highlighted line showing the best result for the shoulder and elbow joints, respectively. FIGS. **29** and **31** show a 3D visualization of grid search values for the shoulder joint (FIG. **29**) and elbow joint (FIG. **31**) over K_P, K_1 and K_D parameters, with the best value being marked. MSE values are shown on a side bar. The values in Table 1 display the results, showing that the elbow joint mainly needed the proportional term while the shoulder joint requires a sufficiently large integral term.

TABLE 1

tuned PID parameters

Parameter	Search Range	Shoulder Best	Elbow Best
K_P	[2, 12]	5.3	9.8
K_I	[0, 4]	1.7	.4
K_D	[0, 2]	0	.1

[0096] The KP, KI, and KD terms of the PID controller were fit to minimize the MSE as described above.

EXAMPLE 4

[0097] This Example describes a reinforcement learning algorithm used in controlling the exemplary robotic arms of Example 1.

[0098] The reinforcement learning control algorithm was adjusted using the robotic arm with perforated joints. The SAC agent was trained in simulation, using a grid search to select hyper-parameters, followed by transfer of the robot skills acquired in simulation to the real robotic system and then fine-tuning the agent on the physical robot. Physical training for the SAC agent occurred only for the perforated joint design on one task (i.e., increasing steps). The SAC agent was subsequently retrained to correct for some errors. The SAC-retrained agent appears in the case of the sinewave task and was fine-tuned on the task to account for the distinct goal dynamics.

[0099] The reinforcement learning problem was defined as maximizing the total discounted future reward in formula (6)

$$R_t=\Sigma_{k=0}^{\infty}\gamma^k r_{t+k} \tag{6}$$

[0100] where γ is a discount factor and rt is the reward given at time-step t. At each time-step, the agent used a policy π(st)=at to select an action at based on the current state st. The state is iterated forward and yields the reward via (st+1, rt)=f(st, at) based on the environment dynamics given by function f. The reinforcement learning algorithm attempts to find the optimal policy, π*(st)=at, which returns the action that maximizes Rt.

[0101] The value function of a state was defined by a formula (7) below:

$$V(s)=\mathbb{E}(Rt|st=s) \tag{7}$$

[0102] while the action-value function was defined as a formula (8) below:

$$Q(s,a)=\mathbb{E}(Rt|st=s,\ at=a), \tag{8}$$

[0103] where the probability distributions of actions are yielded from π(s). Actor Critic algorithms are built to parameterize estimators for the value and policy functions. For the deep reinforcement learning variant, the estimators used were feed-forward neural networks.

[0104] Model-free reinforcement learning techniques such as Proximal Policy Optimization (PPO), Trust Region Policy Optimization (TRPO), Asynchronous Advantaged Actor Critic (A3C) and Soft Actor Critic (SAC) have the capacity to accomplish control tasks in both simulated and physical robotic control tasks. SAC is a sample-efficient and stable reinforcement learning algorithm which follows policy gradient methods, enabling use on continuous action spaces. An entropy maximization formulation encourages exploration and results in a policy that is not overly sensitive to hyper-parameters, making it a prime candidate for the robotic arms of Example 1.

[0105] SAC increases sample efficiency and stability of the training process by increased exploration through rewarding actions that result in higher variance of the policy. This is given by the policy that solves the maximum entropy objective represented in formula (9):

$$\pi^*=\text{argmax}_\pi \mathcal{H}_\pi[\Sigma_{t=0}^{\infty}\gamma(r_t+\alpha\mathbb{E}(\pi(\bullet|s_t)))] \tag{9}$$

[0106] where the expected value of $\mathscr{H}$ defines the entropy of the policy given the current state, and a is a temperature term that scales the impact of the entropy. The entropy term $\mathscr{H}$ is calculated as log π(at|st). Feed-forward neural networks are optimized to minimize the objective functions for the value, Q-value, and policy functions. The corresponding objective functions are listed in formulas (10)-(12), which are alternately minimized until convergence.

$$J_V=\mathcal{H}_{a_t}[½(V(s_t)-\mathcal{H}_{a_t\sim\pi}[Q(s_t,a_t)-\log\pi(a_t|s_t)]^2] \tag{10}$$

$$J_Q=\mathcal{H}_{s_t,a_t}[½(Q(s_t,a_t)-(r(s_t,a_t)+\gamma\mathcal{H}_{s_{t+1}}[V(s_{t+1})]))^2] \tag{11}$$

$$J_V=\mathcal{H}_{s_t}[\mathcal{H}_{a_t}[\alpha\log(\pi(a_t|s_t))-Q(s_t,a_t)]] \tag{12}$$

[0107] Two environments were created to act in parallel, one with the physical robotic arm of Example 1 and one robotic arm created in a MuJoCo simulation as described above in Example 2. An OpenAI gym (available at https://gym.openai.com/) template was expanded to include the custom MuJoCo environment. This allows an agent to receive an observation from an environment, compute, and return an action. The environment can then simulate forward dynamics based on the action, which continued until the task is finished.

[0108] SAC was chosen as the algorithm to update network weights; the network architecture plays roles in the computational capability. Goal-conditioned algorithms allow for accomplishing a variety of tasks through policies that depend on specific goal inputs. Thus, goal-conditioned algorithms may be formulated similarly to controllers that require set points. Since there are delays in the hardware from sensing, communication to the main computer, and other unaccounted for attributes, the current state is non-Markovian. To computationally alleviate this issue, the ten previous observations were included. Both the simulation and physical prototype have a sampling rate of 100 Hz, permitting the history to account for 10 ms delays. One single observation (not including history) contains joint angles θ_x, θ_y recorded from the IMUs, joint velocities ω_x, ω_y calculated from the angles, previous action α_{t-1}, and the goal joint angles g.

[0109] The fidelity of the MuJoCo simulation provided an efficient way to experiment with network hyper-parameters. Multiple hyper-parameter searches were performed to identify efficient learning rate values, quantity of layers, layer sizes, and activation functions. One parameter was varied while the others were held constant, alternating until sufficient values were found. After agents with the varied parameters underwent a training period, mean episode reward was used to determine the best performing hyper-parameters. This process iterated until the mean episodic reward stopped increasing. The final chosen hyper-parameters include dense layers of sizes with ReLU activation functions. Decaying learning-rates α_d outperformed constant learning rates following $\alpha_d(t)=\alpha t/T$ where α is the initial learning rate, t is the current episode, and T is the total number of episodes. The initial learning rate was chosen as $\alpha=0.001$.

[0110] The reward function was incrementally adapted through observation both in MuJoCo and on the physical robot. The final terms for the reward include the distance reward rd and the velocity reward rv. The calculation of reward was done using formulas (13)-(15):

$$r_d=-\sqrt{|\theta_z|+|\theta_y|} \tag{13}$$

$$r_w=-(\sqrt{|\omega_z|+|\omega_y|}) \tag{14}$$

$$r_t=c_1r_d+c_2r_w \tag{15}$$

[0111] The square roots were used since the gradients for updating became increasingly small when the reward was small. This removed a problem of oscillation around the target goal when velocities were very small, yet the distances were still significant. The constants c1 and c2 are weights of the distance and velocity respectively which can help avoid sub-optimal policies—such as a policy which does not move, in order to always minimize velocity. This reward shaping created a dense reward function which trains the agent to minimize distance and reduce action output since the motors hold their position with a control value of 0.

[0112] Three separate tasks were selected to test the effectiveness of the SAC and PID algorithms controlling the robotic arm. Each algorithm receives the current state and goal state as inputs and produces control values for each motor, attempting to match its joint angles and velocities to the proposed goal. FIG. **18** shows a 3D visualization of the end effector's trajectory with its corresponding joint angles, with FIGS. **19** and **20** show angles of the shoulder and elbow joints respectively through the movement of FIG. **19**.

[0113] As shown in FIGS. **21-26**, these physical tasks included the arm following (a) increasing steps (FIGS. **21**

and **22**), (b) decreasing steps (FIGS. **23** and **24**), and (c) sinewave trajectories (FIGS. **25** and **26**). Evaluation of control algorithms on three separate tasks were performed with the shoulder joint (FIGS. **21**, **23**, and **25**) and elbow joint (FIGS. **22**, **24**, and **26**). The goal and experimental trajectories of multiple trials are shown on top, while each the corresponding mean error is shown on bottom. Increasing steps task were used to train SAC, which is most similar to the PID tuning task. Decreasing steps task highlighted asymmetry of joint dynamics. Sinewave task including a SAC-retrained agent that has been fine-tuned to properly capture dynamic goals.

[0114] These experiments were used to determine the ability of both control algorithms to handle smooth and discrete goal-position changes while compensating for gravity, mechanical hysteresis, and other non-linearities. Both joints were evaluated simultaneously, thus errors are dependent on each other. However, it was assumed the measurements are disentangled due to the orthogonality of the actuated axes.

[0115] To see how well each model adapted to new, unfamiliar static and dynamic tasks, the SAC algorithm was trained and the PID controller was tuned on a static task (increasing steps) shown in FIGS. **21** and **22**. For the dynamic sinewave task, a decrease in SAC performance was resolved with a brief retraining period that compensated for the unencountered domain. The PID controller did not require re-tuning because it was tuned about a single point within the range of motion and does not differentiate between static and dynamic tasks.

[0116] In order to train the agent sufficiently, the MuJoCo simulation was used to reduce real-world training time. The training task involved reaching to random points in the state space repeatedly. The most successful agent through training in simulation was transferred and fine-tuned on the physical robot. The physical training task was initially limited to an increasing steps task (FIGS. **21** and **22**) in order to assess generalization. For the sinewave task, fine-tuning of the agent was done by training on increasing steps by briefly continuing training on the new task. Fine-tuning may be used for different joint types, tasks, or simply anything changing the problem while maintaining sufficient similarity.

[0117] Overall, the SAC agent showed lower error for both the elbow and shoulder joints during the increasing steps task (FIG. **21**). The PID controller was only able to perform comparably when operating near the position the control coefficients were tuned for (35°), otherwise having higher errors. The PID controller performs worse on the shoulder joint than the elbow, likely due to uncompensated gravitational forces and the bidirectional mechanical hysteresis depicted in FIGS. **12** and **13**. In contrast, the SAC agent performs consistently throughout each joint's entire range of motion, indicating that the system's transient dynamics have been characterized by the policy network.

[0118] FIG. **23** shows the decreasing steps task, where the SAC agent continued to generalize well. The PID controller failed to minimize the steady state error throughout the range of the experiment, and performs better on the elbow joint compared to the shoulder joint indicating that the elbow is a simpler control task. In both cases, SAC kept reducing in error and the PID controller does not.

[0119] In FIG. **25**, the shape of the error graph (bottom portion) shows a "bouncing" behavior, where the minimal error points represent the time where the true position crosses over the goal trajectory. The PID generalized better than the original SAC for the elbow joint and comparably for the shoulder, explained by its inability to distinguish between task types and more direct control of the elbow. It was observed that the original SAC algorithm had larger error magnitudes. Since the task involved a dynamic trajectory rather than static goal positions, a retrained version of the same algorithm was tested to see if performance improved. The SAC-retrained agent performed significantly better, outperforming the other control methods.

[0120] Error analysis for the SAC and PID algorithms was applied to each joint design. "Shoulder" and "elbow" indicate which physical joint corresponds with the error signal. "Combined" refers to the summation of the shoulder and elbow errors. The background shading of each plot indicates the best performer. The SAC agents outperform the PID controller when considering each combined error. Although SAC does not generalize to the sine wave as well, the SAC-retained algorithm was able to perform best for each joint design. Since all algorithms were fit to the Perforated design, performance generally decreases for the other designs along with larger oscillations.

[0121] Absolute errors of each joint design when undergoing these selected tasks were compared. FIG. **27** shows the error signal of the three trials for each task. The errors are separated by task (increasing steps, decreasing steps, sinewave), joint design (perforated, semi-perforated, solid), and error signal (shoulder, elbow, combined).

[0122] It was observed that the best performing controllers per experiment using the background shading in FIG. **27**, determined by the absolute error summed through time. Overall, the SAC agents perform best in 23/27 cases compared to the PID controller. The results are shown in Table 2. For the most important measurement—the Combined error—SAC outperforms PID control in 9/9 cases. Note that the SAC results include the SAC-retrained.

TABLE 2

Average combined shoulder and elbow error per sample

Task	Design	PID error (°)	SAC error (°)	SAC-r error (°)
Increasing steps	Perforated	1.90	1.55	N/A
	Semi-perforated	2.30	2.27	N/A
	Solid	2.23	2.06	N/A
Decreasing steps	Perforated	2.95	1.76	N/A
	Semi-perforated	4.39	3.85	N/A
	Solid	3.11	2.63	N/A
Sinewave	Perforated	6.77	7.84	3.54
	Semi-perforated	9.26	14.44	8.79
	Solid	6.52	9.23	4.83

[0123] General trends indicate that SAC decreases the amplitude of the error oscillations, lowering the steady-state error through time in the static tasks (top 2 rows of FIG. **27**). Interestingly, the PID control performed best solely with the elbow joint on the sinewave and decreasing steps tasks. In both cases, SAC performs significantly better in the paired shoulder measure, coinciding with how the RL reward function accounts for the sum of both joint errors. Thus, the SAC controller outperformed a PID controller and generalized well through differing physical properties and goal trajectories. Retraining the SAC agent solved a new type of task which also generalized between joint designs.

[0124] The present disclosure provides for a novel method of designing flexible robotic joints, where elastic ligament meshes interweave traditional robotic joints to introduce compliance while maintaining stability. This novel class of robotic joints was used to construct a humanoid arm, demonstrating how the ligament mesh design can be manipulated to adjust dynamic characteristics; thus, three designs were constructed. Robotic joint components (i.e., ball-and-socket and hinge joint components) and molds were designed to accommodate ligaments and 3D printed with negative space to cast a surrounding and interweaving elastic membrane. The simple casting process allows for rapid, low-cost prototyping, opening an avenue for creating systems with dynamic properties through simple mechanical modifications in 3D CAD. The variable nature of this system was accounted for through model-free controllers which successfully control the manipulator across joint designs and tasks.

[0125] Due to both the nonlinearities and variability in fabrication and design parameters, this disclosure compared two model-free control algorithms: the SAC reinforcement learning algorithm and a PID controller. The PID parameters were chosen through a grid search optimizing for MSE values. Since RL algorithms require large training times, a simulation was used to determine the SAC architecture through a grid search of hand-picked network hyper-parameters including learning rate, hidden layer quantity, hidden layer size, and activation function. The best simulated agent was transferred to the physical robot and trained on a single task, drastically reducing training time on the robot.

[0126] Requisite robot training hours were replaced with simulation training (e.g., training a model which generalizes with varied dynamic properties) and subsequent transfer of the robot skills acquired in simulation to the real robotic system. The SAC algorithm was only trained on one joint design (i.e., perforated) and one task (i.e., increasing steps), yet generalizes well to each experiment involving static goal positions. Fine-tuning the agent on the dynamic task (i.e., sinewave) took little time and generalized for each joint design. Examining the performance of the manipulator built from each joint design (i.e., perforated, semi-perforated, solid) through each task (i.e., increasing steps, decreasing steps, sinewave) shows that SAC outperforms the PID controller (FIG. **27**).

[0127] Overall, the differences are evident between the two control methodologies. Although the PID controller achieves satisfactory results on the tuned region, the control accuracy diminished as the distance from the region increased. This is most prominent in the shoulder joint, and with tasks requesting a decrease in angle. The shoulder joint experiences a significantly larger moment of inertia along with compounding nonlinear dynamics which the PID controller struggles to account for. This indicates that the PID controller is sufficient for simple tasks such as controlling a one-joint one-link mechanism. Thus, fine-tuning a simulation-trained reinforcement learning agent provides a method for solving more complex dynamic problems.

[0128] Regarding control of these joints, reinforcement learning has the potential to generalize in the case of complicated joint-link combinations and varying dynamics. Use of SAC algorithm achieved great results. It is envisioned that additional modifications may be used to speed up training and expand generalization such as: Hindsight Experience Replay (HER), domain randomization, actuator randomization, curriculum learning, meta-learning and more. Since this manipulator is similar to the human arm, there is potential to use motion capture data, enabling techniques which use expert demonstrations such as apprenticeship learning and generative adversarial imitation learning (GAIL).

[0129] While the present disclosure only provided three different joint designs (i.e., perforated, semi-perforated, and solid) various other designs are contemplated. Characterization of how each design parameter directly influences the dynamics could yield programmable soft-rigid joints. The design of the manipulator can be sophisticated and refined to provide additional primary degrees of freedom and more exact motion. Also, while the involved tasks (i.e., increasing steps, decreasing steps, sinewave) are representative of static and dynamic goals, they are not exhaustive and other tasks suitable for developing control agents are contemplated. A study examining tasks which best train an agent across the whole state-space would highlight the advantages provided by the proposed training pipeline.

[0130] Furthermore, a strength of the joint design is that it aids load bearing tasks while diminishing external disturbances. Such tasks could involve adapting to varying loads strictly through observing the changed dynamics or performing high-level tasks such as object manipulation. Currently the proposed manipulator uses gravity and spring force to open the joints, yet antagonistic motors are biological and grant tunable compliance. In human arms, injuries show obvious modifications of strain curves which influences bio-mechanical properties, and the proposed system may be used to generate similar modifications. This disclosure provides a platform to produce a variety of robot arms mechanically akin to healthy/injured human arms, damaged/fatigued iterations, limbs with desired joint properties, and more. Furthermore, this disclosure provides foundational steps leading to a future where users can design a task, and simulations optimize the design and creation of a robot which best serves the task.

[0131] This disclosure also verified the ability to adjust the joint dynamics through varying the quantity of through-holes in the elastic membrane of each joint, experimentally testing three conditions: perforated, semi-perforated, and solid. The through-holes act as parallel spring-damper systems, effectively adjusting the non-linear dynamics. Experimentally the disclosure analyzed the stiffness in FIGS. **10-13**, and the damping in FIGS. **14-17**. Non-linearities manifested primarily in the form of hysteresis, time delays, and higher order functions. The results clearly support a need for data-driven non-linear estimators, and the parameters found proved to be sufficient for simulations which inform learning algorithms.

[0132] It will be appreciated that of the above-disclosed and other features and functions, or alternatives thereof, may be desirably combined into many other different systems or applications. Also, that various presently unforeseen or unanticipated alternatives, modifications, variations or improvements therein may be subsequently made by those skilled in the art which are also intended to be encompassed by the following claims. Unless specifically recited in a claim, steps or components according to claims should not be implied or imported from the specification or any other claims as to any particular order, number, position, size, shape, angle, or material.

What is claimed is:

1. A robotic arm comprising:

a first link;

a second link;

a joint interconnecting the first link and the second link, such that the first link is movable relative to the second link along an axis of motion, the joint including:

a socket component coupled to a distal end portion of the second link;

a ball component coupled to a proximal end portion of first link, the ball component configured to rotationally fit within the socket component; and

a flexible membrane encasing the socket component and the ball component.

2. The robotic arm according to claim **1**, wherein the socket component includes a first plurality of through-holes.

3. The robotic arm according to claim **2**, wherein the ball component includes a second plurality of through-holes.

4. The robotic arm according to claim **2**, wherein the first plurality of through-holes is disposed in a first socket plane and a second socket plane.

5. The robotic arm according to claim **4**, wherein the first socket plane and the second socket plane are perpendicular to each other and are aligned with the axis of motion.

6. The robotic arm according to claim **3**, wherein the second plurality of through-holes is disposed in a first ball plane and a second ball plane.

7. The robotic arm according to claim **6**, wherein the first ball plane and the second ball plane are perpendicular to each other and are aligned with the axis of motion.

8. The robotic arm according to claim **3**, wherein the flexible membrane is disposed within a space defined by the first plurality of through-holes and the second plurality of through-holes.

9. The robotic arm according to claim **1**, wherein the flexible membrane is formed from an elastomer.

10. The robotic arm according to claim **1**, further comprising an actuator coupled to the second link and a cable coupled to the first link, wherein the actuator is configured to move the first link by spooling the cable.

11. The robotic arm according to claim **10**, wherein each of the socket component and the ball component includes at least one routing block configured to route the cable around the joint.

12. A method of manufacturing a robotic joint, the method comprising:

forming a first plurality of through-holes in a ball component;

forming a second plurality of through-holes in a socket component; and

inserting the ball component into the socket component to form a robotic joint, wherein the ball component is movable relative to the socket component along an axis of motion.

13. The method according to claim **12**, further comprising:

applying a flexible membrane over the robotic joint and within a space defined by the first plurality of through-holes and the second plurality of through-holes.

14. The method according to claim **13**, wherein applying the flexible membrane includes placing the robotic joint in a mold and pouring a liquid precursor composition of the flexible membrane.

15. The method according to claim **12**, wherein the first plurality of through-holes is disposed in a first socket plane and a second socket plane and the first socket plane and the second socket plane are perpendicular to each other and are aligned with the axis of motion.

16. The method according to claim **12**, wherein the second plurality of through-holes is disposed in a first ball plane and a second ball plane and the first ball plane and the second ball plane are perpendicular to each other and are aligned with the axis of motion.

17. A method for programming a control agent for controlling a robotic arm, the method comprising:

running a simulation of a robotic arm based on a plurality of physical parameters on a workstation;

training a reinforcement learning algorithm based on the simulation of the robotic arm; and

loading a control agent including the reinforcement learning algorithm into a controller controlling the robotic arm.

18. The method according to claim **17**, wherein training includes:

performing a plurality of tasks to reach a random point a three-dimensional space of the simulation.

19. The method according to claim **18**, wherein training further includes:

implementing a reward function configured to minimize distance traveled to the random point.

20. The method according to claim **17**, further comprising:

retraining the reinforcement learning algorithm based on operation of the robotic arm in a physical space.

* * * * *