ARTICLES
https://doi.org/10.1038/s42256-019-0029-0
nature machine intelligence

Autonomous functional movements in a tendon-driven limb via limited experience

Ali Marjaninejad[1,2], Darío Urbina-Meléndez[1], Brian A. Cohn[3] and Francisco J. Valero-Cuevas[1,2,3,4,5*]

Robots will become ubiquitously useful only when they require just a few attempts to teach themselves to perform different tasks, even with complex bodies and in dynamic environments. Vertebrates use sparse trial and error to learn multiple tasks, despite their intricate tendon-driven anatomies, which are particularly hard to control because they are simultaneously non-linear, under-determined and over-determined. We demonstrate—in simulation and hardware—how a model-free, open-loop approach allows few-shot autonomous learning to produce effective movements in a three-tendon two-joint limb. We use a short period of motor babbling (to create an initial inverse map) followed by building functional habits by reinforcing high-reward behaviour and refinements of the inverse map in a movement's neighbourhood. This biologically plausible algorithm, which we call G2P (general to particular), can potentially enable quick, robust and versatile adaptation in robots as well as shed light on the foundations of the enviable functional versatility of organisms.

Today's successful control algorithms for robots often require a combination of accurate models of the physical system, task and/or the environment or expert demonstration of the task, as well as expert knowledge to adjust parameters or extensive interactions with the environment[1–12]. Even then, many rely heavily on error corrections via real-time state observation or error feedback[3,4,8,9,13–21]. Moreover, some prefer to focus on simulated behaviour of simplified systems and environments or limit the physical system to simple scenarios (for example, only kinematic control)[7,15,16,22–29]. Although advances in machine learning demonstrate that reinforcement learning (RL) agents can achieve human-like performance in complicated tasks (for example, video games) or can find optimal strategies for mechanical tasks using evolutionary algorithms, those studies are limited to computer simulations due to the numerous attempts needed for the algorithm to converge[30–32]. In addition, some researchers are seeking to apply biologically plausible principles from anatomy and neuroscience to develop versatile robots and learning strategies[3,6,7,18,20,25,26,33–35]. In particular, there is a need to develop feed-forward, model-free approaches that learn using limited interactions with the environment (that is, 'few-shot' learning[36]), which could imbue robots with the enviable versatility, adaptability, resilience and speed of vertebrates during everyday tasks[4,10,37–39].

This work presents a combination of hardware and software advances (in contrast to much current work in robot learning, which is carried out only in simulations) that demonstrate how a model-free, open-loop approach allows few-shot autonomous learning to produce effective movements in a three-tendon two-joint limb. Moreover, our approach (Figs. 1 and 2) is biologically plausible at two levels: (1) we use motor babbling—as do young vertebrates[40,41]—to learn the general capabilities of the physical systems (also called 'plant' in control theory), followed by reinforcement of high-reward behaviour and refinements that are particular to the task (that is, general-to-particular, or G2P); (2) we use tendons to generate torque at each joint (Fig. 3 and Supplementary Fig. 1) to replicate the general problem that biological nervous systems face when controlling limbs[42] (which makes for a simultaneously over- and under-determined control problem; see Methods). This may lead to a class of robots with unique advantages in terms of design, versatility and performance[43]. This work also contributes to computational neuroscience by providing a biologically and developmentally tenable learning strategy for anatomically plausible limbs (Supplementary Discussion).

Results

We show that the G2P algorithm can autonomously learn to propel a treadmill (while supported by a carriage) without closed-loop error sensing or an explicit model of the dynamics of the tendon-driven limb or the environment (for example, limb inertia, contact dynamics or expected reward). We also show that the execution of multiple attempts can itself lead to improvement in performance on account of a refined inverse map in the neighbourhood of the movement. Such cost-agnostic improvements serve as a proof of principle of a biologically tenable mechanism that benefits from familiarity with the task, rather than teleological optimization or even error-driven corrections.

Results for cyclical movements to propel the treadmill. A given run begins with a 5 min motor babbling session where the time history of a pseudo-random control sequence (a three-dimensional (3D) time-varying vector of step changes of the current to each motor) is fed to the limb while its kinematics (joint angles, angular velocities and angular accelerations) are measured by encoders at each joint (Fig. 1 shows an overview of G2P). An artificial neural network (ANN) then uses these motor babbling data to create an initial inverse map from 6D kinematics to a 3D control sequence. A movement to propel the treadmill is parameterized by a closed orbit in 2D joint-angle space that interpolates between the 'feature vector' of 10 evenly distributed points (Fig. 1c). For a given cycle duration of ~1 s, this defines the 6D limb kinematics: joint angles, angular velocities and angular accelerations for each of the two joints (see

[1]Department of Biomedical Engineering, University of Southern California, Los Angeles, CA, USA. [2]Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA, USA. [3]Department of Computer Science, University of Southern California, Los Angeles, CA, USA. [4]Department of Aerospace & Mechanical Engineering, University of Southern California, Los Angeles, CA, USA. [5]Division of Biokinesiology & Physical Therapy, University of Southern California, Los Angeles, CA, USA. *e-mail: valero@usc.edu

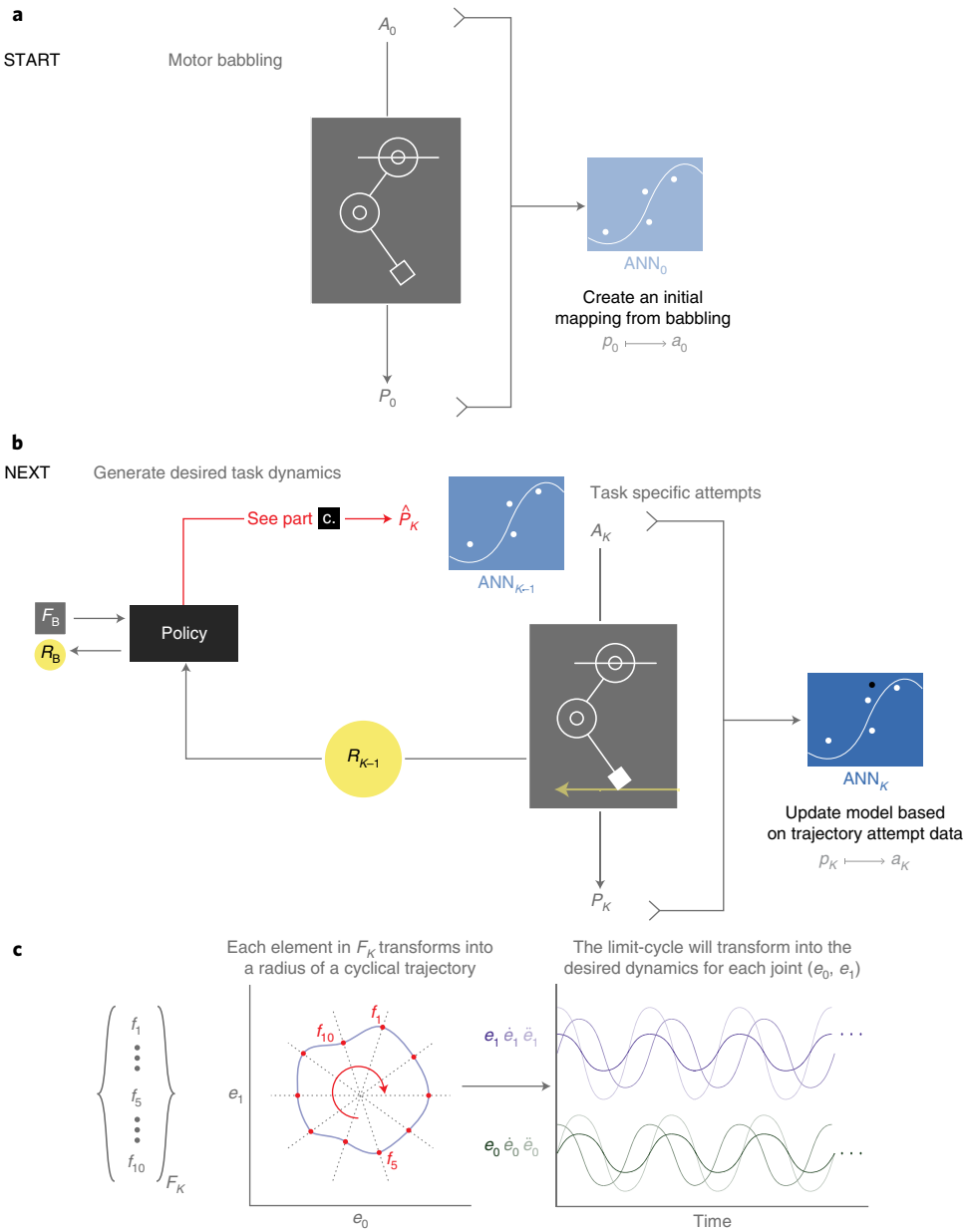

Fig. 1 | The G2P algorithm. a, Every run of the algorithm begins with time-varying babbling control sequences (activations A_0, which run through the electric motors) that generate 5 min of random motor babbling (P_0). These input–output data are used to create an inverse (output–input) map ANN_0 from limb kinematics to control sequences. **b**,**c**, RL begins by varying the ten free parameters of the feature vector (F_K) (**b**), defining a cyclical movement (**c**). These movements can, in principle, propel the treadmill. ANN_0 maps each candidate desired kinematics (P_K) into the activation sequences (A_K), which propel the treadmill (where P_K is the resulting kinematics) and yield a reward (R_K). An attempt (where K is the attempt counter) is when an activation sequence is repeated 20 times and used to produce 20 steps worth of kinematic data. These kinematic data are further processed and concatenated with all prior data to refine the inverse map into ANN_K. The total treadmill propulsion, if any, is the reward for that attempt. The system remembers the best reward so far, and the feature vector that generated it. If a new feature vector yields a better reward, the memory will be updated. The system will continue its search in an increasingly smaller neighbourhood of that feature vector (that is, the search neighbourhood gets smaller as the reward increases) and send the resulting kinematics to the ANN to further refine the inverse map. However, note that data from all attempts (whether they improve on the best so far or not) are used to refine the inverse map. Figure 2 describes data processing for each run.

Methods for details). Next, 20 replicates of these kinematics are fed through the initial inverse map (lower-level control), which produces 20 cycles of a control sequence (Fig. 1c). Those control sequences are delivered to the robotic limb to produce 20 cycles. The reward for that attempt is a scalar value representing the distance the treadmill was propelled backward, in millimetres, as in forward locomotion. Reward for each attempt is provided to the system in a discrete way (only after the attempt—20 cycles—is over).

A sequence of attempts (Fig. 2) within each run of the G2P algorithm (Fig. 1) uses the initial inverse map to start the exploration phase: the ten free parameters of the feature vector are changed at random and the resulting dynamics are sent to refine the inverse map. The resulting control sequence is fed to the motors to produce limb movement until the treadmill reward crosses a threshold of performance set to 64 mm (empirically selected to lead to clearly observable propulsion). Thereafter, the exploitation phase of

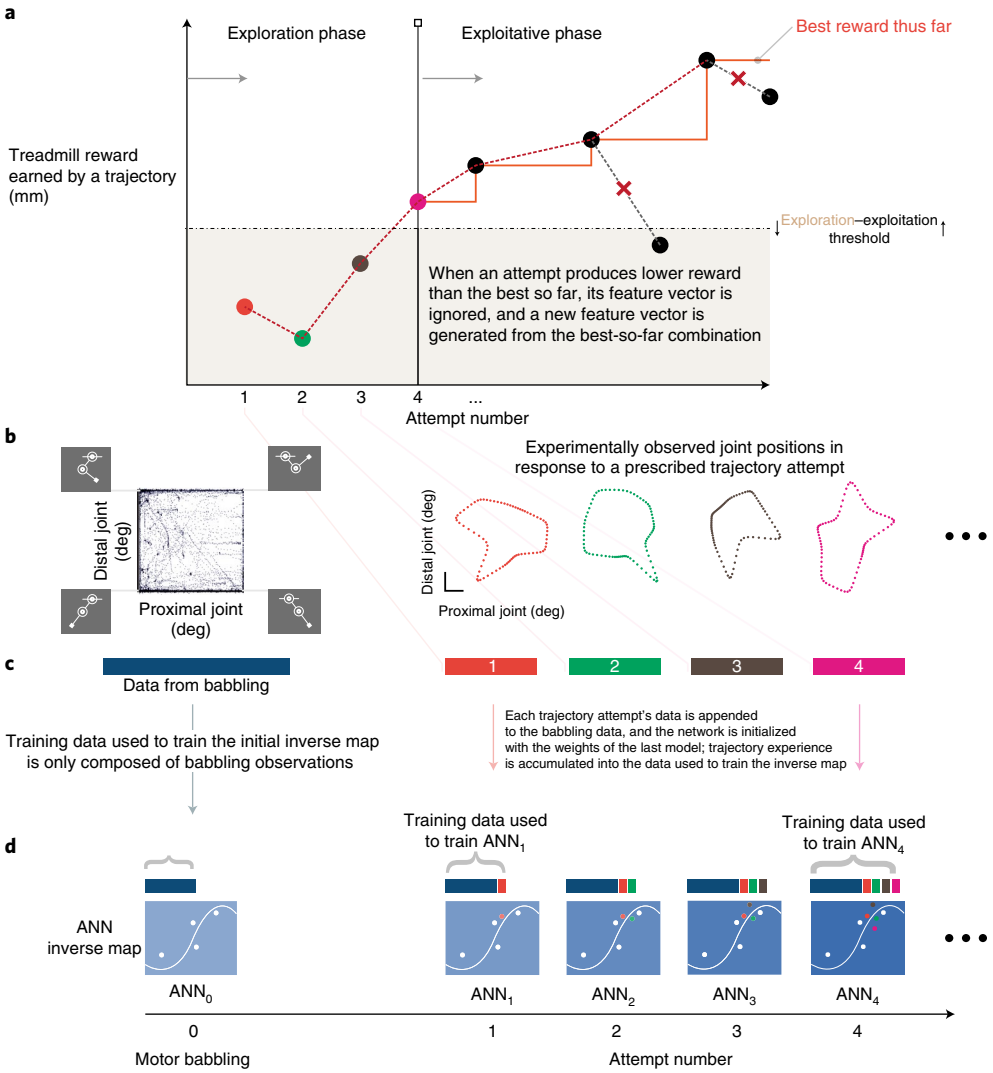

Fig. 2 | A run of the G2P algorithm, in detail, for the reward-driven treadmill task. a, Evolution of reward across the exploration and exploitation phases. The exploration phase begins by using the initial inverse map ANN_0 (Fig. 1) to attempt to produce the cyclical movement defined by the first feature vector selected from a random uniform distribution. The predicted control sequence is applied to the motors to produce 20 cycles of movement that yield a particular treadmill reward (orange dot) and continues to be changed until a feature vector is found that yields a reward above the exploration–exploitation threshold (dotted line). It then transitions to the exploitation phase where the feature vectors of the subsequent 15 attempts are sampled from a 10-dimensional Gaussian distribution centred on the best feature vector so far. **b**–**d**, Motor babbling and sequential task-specific refinements of the inverse map: distribution of the proximal and distal joint data from motor babbling (enlarged in Fig. 6) and subsequent attempts (colour coded) (**b**); babbling data (shown schematically as a blue bar) were used to generate the initial inverse map (ANN_0) (**c**) and concatenated with data from each attempt to continually refine the inverse map (ANN_1, ANN_2, …) (**d**).

G2P begins: we use policy-based RL with stochastic policy search in which the feature vector is sampled from a 10D multivariate Gaussian distribution. The mean vector of this Gaussian distribution is the best feature vector (that is, that yielded the highest reward so far), and its standard deviation (s.d.) values shrink as the reward increases (see Methods). Feature vectors sampled from this Gaussian are used in subsequent attempts. Those that produce higher reward serve as the new best feature vector (see Methods for more details). This process resembles an evolutionary algorithm and is similar to a cross-entropy optimization method, with the distinction that here we use just one candidate solution (as opposed to a population of solutions) and the s.d. is a function of the reward (as opposed to the s.d. of the subpopulation with the highest rewards). Each time a control sequence is applied (in either the exploration or exploitation phase), the resulting kinematics are recorded, appended to the babbling data and any prior attempts, and included in the next refinement of the inverse map (Fig. 2b). That is, every interaction with the physical system is used in the next attempted refinement of the inverse map. This is analogous to trial-to-trial experiential adaptation during biological motor learning[40].

Figure 4a shows the reward for each sequential attempt for 15 independent runs labelled A–O. These colour-coded stair-step lines show the best reward achieved thus far. Our system was able to cross the exploration–exploitation threshold in a median of 24 attempts, and the subsequent exploitation phase showed a median reward improvement of 45.5 mm with a final reward median of 188 mm (best run performance was 426.9 mm). Simulation results for the corresponding test are shown in Supplementary Fig. 2.

Figure 4b shows that the system is able to learn families of related solutions (that is, a motor habit), and that—for each such family—high rewards can be achieved with both high and low power consumption. This shows that energy minimization is not an

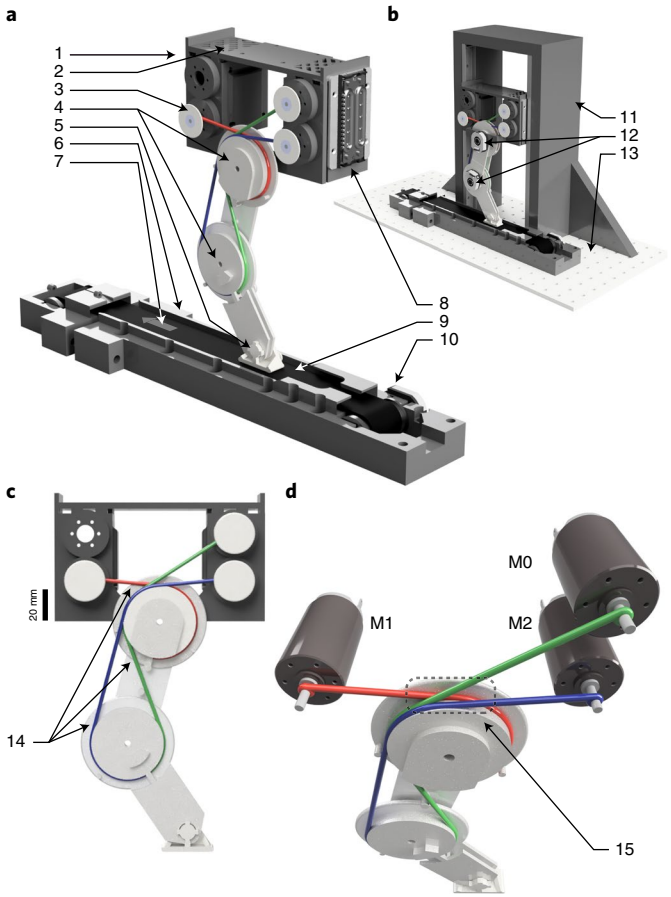

Fig. 3 | Planar robotic tendon-driven limb. a, General overview of the physical system: (1) motor-joint carriage, (2) motor ventilation, (3) shaft collars, (4) joints (proximal and distal), (5) passive hinged foot, (6) treadmill, (7) direction of positive reward, (8) linear bearings on carriage (locked at a particular height during testing), (9) treadmill belt, (10) treadmill drum encoder. **b**, Fully supported system: (11) frame, (12) absolute encoders on proximal and distal joints, (13) ground. **c**, Tendon routing: (14) three tendons driven by motors M0, M1 and M2. **d**, System actuation. Motor M1 drives only the proximal joint anticlockwise, while M0 and M2 drive both joints (M0 drives the proximal joint clockwise, and the distal joint anticlockwise, while M2 drives both joints clockwise). (15) Tendon channel.

emergent property in this biologically plausible system or learning strategy. However, if desired, an energy optimization term could be appended to the reward to yield this property.

Results for free cyclical movements in air. The utility of familiarity with a task to produce incremental improvements (by increasing the precision of the inverse map) cannot be directly interpreted from the results in Fig. 4. This is because the RL algorithm might, by itself, find a feature vector that yields high reward even with an imprecise inverse map. However, in many applications, such as tracking a desired trajectory (a form of imitation), precision of this inverse map is crucial. We therefore performed two trajectory-tracking tasks in air (with no explicit reward or real-time feedback) to evaluate the performance of G2P in refining the inverse map during task-specific explorations for a given cyclical trajectory as well as the generalizability of these refinements on unseen cyclical trajectories.

Task A: Free cyclical movement in air for a single trajectory. The limb was suspended 'in the air' without making contact with the treadmill while, as before, the initial inverse map was extracted from 5 min of motor babbling data. For each run, this initial inverse map (ANN_0) was incrementally refined with data from each of five attempts, regardless of its tracking error over the course of the attempt. Figure 5a,i shows reduction of the mean square error (m.s.e.) with respect to the attempt number for one sample run. Figure 5a,ii,iv shows the time history of actual achieved versus desired joint angles for those same five attempts (see Supplementary Fig. 3a,b for the simulation results of the corresponding test). Supplementary Fig. 4 also shows the boxplots of the number of iterations for babbling and the following four refinements over 50 replicates using data recorded from the physical system during this task.

Task B: Generalizability of learned free cyclical movements in air. Although we have demonstrated how repeated exposure to the same task improves the performance of that task (Task A and Fig. 5a), this does not speak to the generalization of a given inverse map to the execution of other unseen trajectories. Here, we followed motor babbling with serial refinements over 30 randomly selected trajectories (features sampled from a uniform distribution within the 0.2–0.8 range). The trained inverse map was then 'fixed' and evaluated for its m.s.e. accuracy on 30 additional unseen random (same random distribution) trajectories (the test set) without further refinement. Figure 5b,i,ii shows that this refined inverse map performed better on the test set. This strongly suggests that refining a map with specific examples improves performance on a variety of test tasks and does not over-fit to its training set. As such, the refined map captures well the complex mechanics of the tendon-driven double-pendulum limb to produce dynamic cyclical movements. This is very important as it means G2P can learn from every experience and generalize it to similar tasks (see Supplementary Fig. 3c for the simulation results for the corresponding test). The fact that we stack all data (babbling and every new experience) to refine the ANN enables the system to improve the performance for other related tasks without forgetting the old ones (see Methods).

Robustness to perturbation. In a variant of test A (after babbling and 10 refinement attempts), we struck the limb with a metal rod once the system was moving at steady state. This blunt perturbation pushed the limb away from its cyclical movement, but the system then returns to its steady-state behaviour after ~1 cycle (Supplementary Video 2). Poincaré return maps and stability analysis[44] for these perturbation tests are provided in the Supplementary Information (Supplementary Figs. 5 and 6).

Point-to-point and more complex non-cyclical movements. For the point-to-point tests, the system starts at an initial posture and then performs ramp-and-hold transitions to each of five different positions in the joint angles space. For the complex, non-periodic task, the system is instructed to follow a non-periodic trajectory for each joint. Each of these trajectories consists of smooth and ramp-and-hold movements (both in-phase and out-of-phase) of each joint (although the other joint might be moving). This is particularly challenging because two of the tendons cross both joints, so isolated movement of one joint requires coordination across all tendons. Supplementary Video 2 shows an instance of each of these tests. The system (which operates open-loop) performed both tasks reasonably. Supplementary Fig. 7 presents the results. Although the system's performance for arbitrary and more complex movements needs to be investigated further, these results serve as an encouraging proof of principle that extends the utility of the G2P algorithm beyond cyclical movements—the focus of this first investigation.

Discussion

The G2P algorithm produced two important results in the context of the challenging task of few-shot learning of feedforward

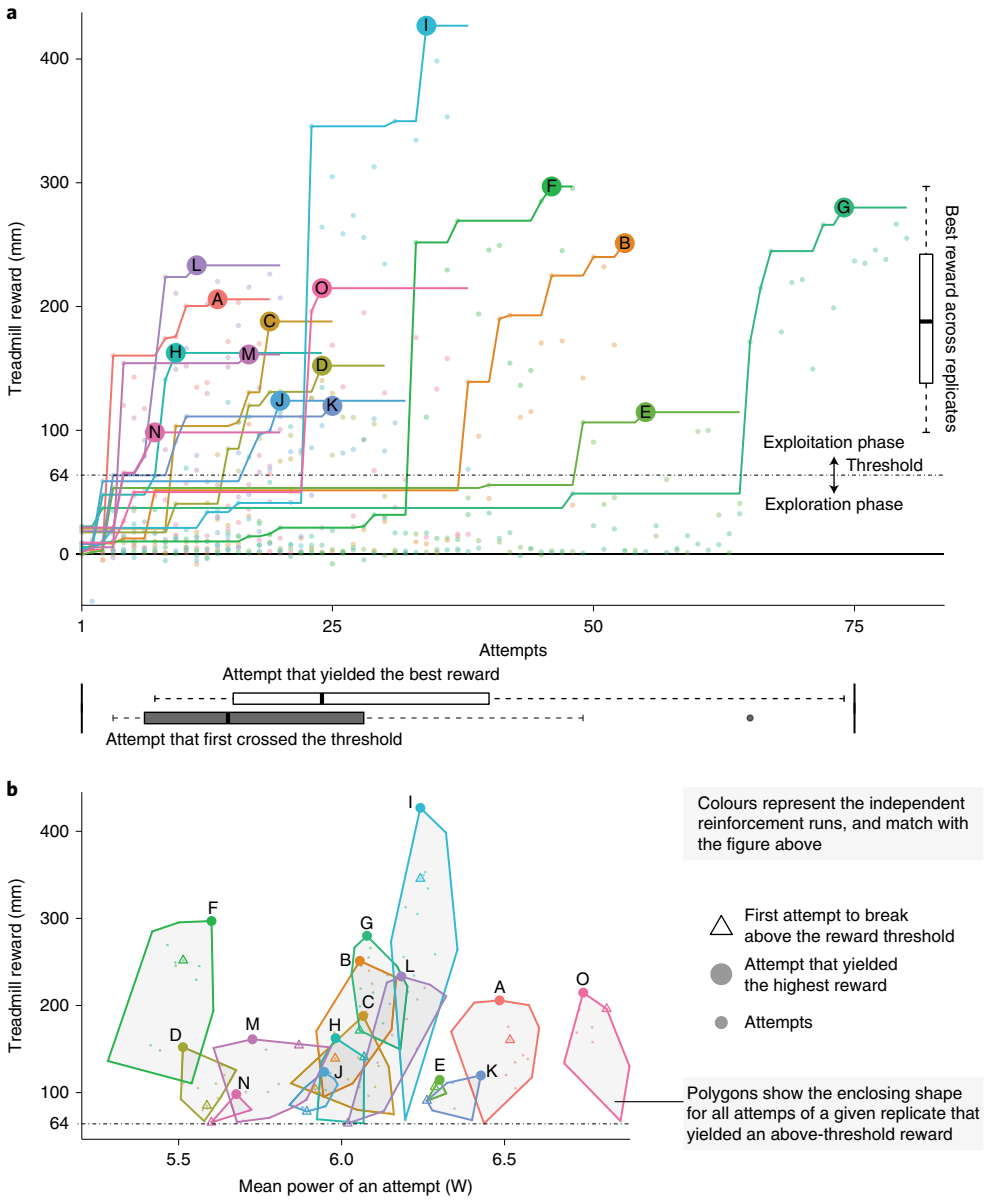

Fig. 4 | Treadmill task results. a, Treadmill reward accrued in each of 15 independent runs, labelled A–O: all runs crossed the exploration–exploitation threshold of 64 mm of treadmill propulsion (median of exploration attempts, 15). All runs showed improvement, and the median number of attempts needed to reach the best reward of each run was 24. **b**, Reward versus energy consumption (mean power of an attempt): the plot shows all attempts from runs that garnered a reward above the exploration–exploitation threshold on the reward versus energy consumption plane. We can then find the convex hull representing them as a family of similar solutions, or a motor habit. For each polygon, the peak reward (large circle) and the reward from the first attempt to cross the threshold (triangle) are shown. We detect no right-to-left trend, indicating that energy consumption was spontaneously reduced as performance improved. Conversely, higher reward did not always require higher energy consumption, even though more external work was being done to propel the treadmill the furthest.

and robust production of a cyclical movement of a tendon-driven system. This brings novel possibilities to robotics in general as it shows that a few-shot approach to autonomous learning can lead to effective and generalizable control of complex limbs for movements and, by extension, a new generation of biologically plausible robots for locomotion, manipulation, swimming and flight. Given its biologically tenable features, G2P can ultimately also enable the control of neuromorphic systems (for example, ref. [45]) to help explain the versatility of neuromuscular systems.

How does G2P relate to the field?. The G2P algorithm's main contribution is that it combines developmentally and biologically plausible approaches in both hardware and software to autonomously learn to create functional habits that produce effective feedforward behaviour—where familiarity reinforces habits without claim to uniqueness nor global optimality. Moreover, it does so based on a data-driven approach that uses few shots (that is, limited experience) seeded by motor babbling. Importantly, it does so in the physical world for a biologically plausible tendon-driven limb for complex dynamic tasks with and without intermittent contact, and not just in simulation. We now discuss how this novel integrative approach compares and contrasts with other work in machine learning, reinforcement learning and control theory.

We used a model-free approach because precise prior knowledge of the system and the environment is not usually available for dynamic tasks in the physical world[4,8,10,37,38]. This is also the case for

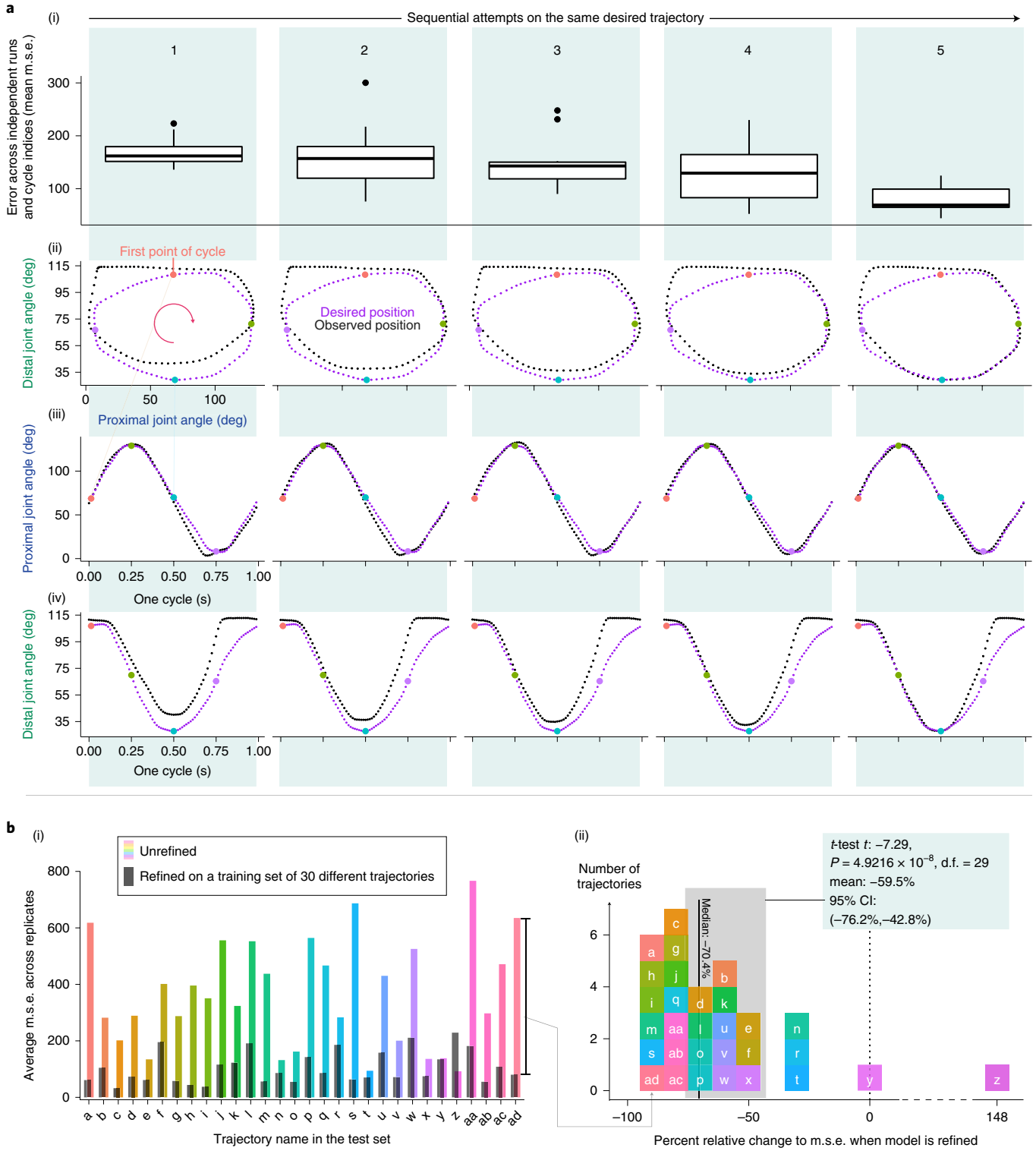

Fig. 5 | A run of the G2P algorithm in detail for the tracking of free cyclical movements. a, Improvements in performance resulting from five attempts at producing a target cyclical movement defined by a given feature vector. i, boxplots of m.s.e. ii–iv, Desired versus actual joint kinematics. **b**, Test of generalization of refined model over unseen trajectories a, b, …, ad (see main text). i, m.s.e. of the 30 test trajectories executed using either the unrefined inverse map (only babble-trained, colour bars) or the refined inverse map (sequentially over 30 other training trajectories, grey bars). ii, Histogram of percent difference in m.s.e. for the results in i for each of the 30 unseen test trajectories. CI, confidence interval.

systems that rely on experts to manually tune system parameters, select the appropriate hyper-parameters or provide demonstrations of the task[4,6–8,11,14]. Without such knowledge, the system often needs to execute numerous iterations in the real world, simulation (real-time or offline) or both to converge on adequate performance, which can make the learning process costly[2,10,12,28–32]. Therefore, data-driven model-free systems that do not rely on prior knowledge and can learn with minimal experience are needed[10,37,38].

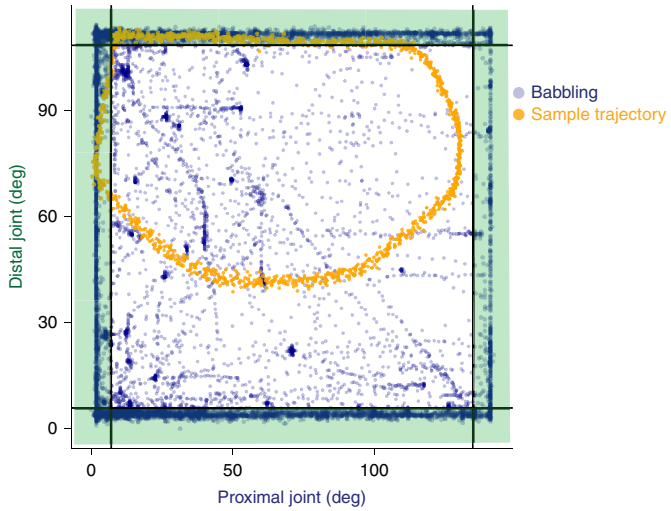

Fig. 6 | Distribution of joint angles visited during motor babbling versus those used to produce a free cyclical movement in air. Motor babbling is done under no supervision, and in this tendon-driven double pendulum primarily results in movements that rapidly fly towards the extremes of the ranges of motion of each joint (84.3% lie in the shaded region within 5% of the joint limits (black lines)). In contrast, the desired movement trajectories require exploitation of the relatively unexplored internal region of the joint angle space (orange points represent 15 repeated cycles of a given cyclical movement).

A common approach in robotics today is a compromise: use models of a system to first develop controllers in simulation (for example, refs. [1,2,17]) and then deploy them in physical systems (often known as transfer learning).

Feedback can play an essential role in biological or engineering control. At times, however, feedforward systems can be advantageous. This is especially the case when real-time computation is not available, the state cannot be observed reliably or when delays are large compared to the dynamics of the task[10]. Thus, a real-time feedback system can be costly for engineered and biological systems[4]. Alternatively, feedforward control using precise inverse maps can be used to minimize the reliance on feedback. Therefore, an efficient system should only utilize feedback when necessary. In fact, this is even the case in biological systems where, for example, movement-related sensory feedback is not necessarily needed for humans to learn to execute a motor skill[39].

Adequate performance in the physical world is a desirable property for any controller, as it demonstrates its robustness to the full set of dynamics and disturbances. Successful control of tendon-driven limbs in real-world physics is a challenging test of learning and control strategies[2,18,27,42,43]. Roboticists find such anatomies particularly hard to control because they are simultaneously nonlinear, under-determined (many tendon tensions combine to produce few net joint torques) and over-determined (few joint rotations define how many tendons need to be reeled in/payed out)[42,43]. Some have successfully controlled such tendon-driven systems in the real world using feedback control of fingers[18] and manipulation[2]. Others have used simulations to produce simple tasks (hopping/point-to-point movements through manual tuning of parameters[7]). Our work is a real-world demonstration of autonomous learning for feedforward control of dynamic cyclical and discrete tasks in a tendon-driven system via few-shot learning and minimal prior knowledge.

Familiarity reinforces habits. Motor babbling creates an initial general map from which a control sequence for a particular movement is extracted. This initial prediction serves as a 'belief' about the relationship between the body and environment, and an appropriate control strategy. This prediction is used for the first attempt, which, although imperfect, produces additional sensory data in the neighbourhood of a particular task. These data are subsequently leveraged toward refinement of the inverse map, leading to an emergent improvement in performance and reinforcement of useful beliefs.

Importantly, the details of a given valid solution are idiosyncratic and determined by the first randomly found control sequence that crossed the exploration–exploitation threshold of performance (Fig. 4). Hence all subsequent attempts that produce experience-based refinements are dependent on that seed (much like a Markov process). This solution and its subsequent refinements, in fact, are a family of related solutions that can be called a 'motor habit' that is adopted and reinforced even though it has no claim to uniqueness nor global optimality[46]. Biologically speaking, vertebrates also exhibit idiosyncrasies in their motor behaviour, which is why it is easy to recognize health states, sexual fitness, identify individuals by the details of their individual movement and speech habits, and even tell their styles and moods. A subtle but important distinction is that these emergent motor habits are not necessarily local minima in the traditional sense. They are good enough solutions that were reinforced by familiarity with a particular way of behaving. There is evidence that such multiplicity of sub-optimal, yet useful, set points for the gains in spinal circuitry facilitates finding solutions to produce discrete and cyclical movements[46]. Those authors argue that it is evolutionarily advantageous for vertebrates to inherit a body that is easy to learn to control by adopting idiosyncratic, yet useful, motor habits created and reinforced by an individual's own limited experience, without consideration of global optimality[46]. G2P uses a similar learning strategy.

Figure 5a also demonstrates familiarity as an enabler of learning, where we tested the ability to produce free cyclical movements in air, without contact with the treadmill—and hence without explicit reward. The performance of a particular free cyclical movement improves simply on the basis of repeated attempts. This represents essentially the cementing of a motor habit on the basis of experience in the neighbourhood of the particular movement. Figure 6 further shows 15 cycles of a particular free movement in the interior of the joint angle space, even though it is the most poorly explored region during babbling. Importantly, familiarity with the neighbourhood of a task need not lead to over-fitting that is only locally useful. Our cross-validation experiments in Fig. 5b show that familiarity with one's motion capabilities for some tasks seems to inform the execution of other tasks. Note that the absence of a reward or penalty for particular joint angles allowed the emergent solution to contain a portion where the distal joint is at its limit of range of motion. This, however, need not be detrimental to behaviour. For example, human walking often has the knee locked in full extension immediately before heel strike.

Task reward versus energetic cost. Studying whether energetic efficiency during locomotion is an emergent property, or must be actively enforced, is a longstanding question in motor control[47–49]. The results in Fig. 4b are particularly interesting because they show that energy minimization is not an emergent property in this system[50]. Figure 4a shows the sequence of attempts from each run. Each family of attempts that perform above the exploration–exploitation threshold (plotted as the polygonal convex hull that includes them; Fig. 4b) can be narrow or wide from the perspective of energetic cost (horizontal axis), but nowhere do we see a general trend towards energy minimization within families (that is, none of the convex hulls are shaped diagonally towards the top left). Conversely, one could have expected that movements that caused more propulsion would be more energetically costly as they do more mechanical work against the treadmill, yet we also do not see such a consistent trend diagonally towards the top right. This is not to say that the

high-level controller can add energy minimization as an element of the cost—although it may jeopardize the ability of the limb to apply mechanical work to the treadmill. Energy consumption may be necessary to regulate dynamic tendon shortening and lengthening (that is, internal strain energy) to produce proper kinematics—a consequence of the simultaneously over- and under-determined nature of tendon-driven limbs[42].

Limitations, opportunities and future directions. For organisms, as for machines, there is a tradeoff between improving performance via practice as each attempt carries the risk of injury, fatigue and wear of tissues (for example, blisters, inflammation of tendons, stress fractures)—in addition to energy expenditure and opportunity cost (that is, spending time refining one task precludes learning a different one in a zero-sum lifespan). The G2P algorithm is designed to yield reasonable—if suboptimal—performance with limited data and no real-time feedback, but where the system continues to learn from each execution of the task. However, it is also amenable to goal-driven refinements as each solution can serve as a starting point for subsequent optimization or improvements through feedback-driven corrections (proportional–integral–derivative, recurrent neural networks and so on)[51–55].

Our fundamental motivation is to replicate how biological systems learn to move in a well-enough fashion when they must also limit the number of attempts using their own bodies. Our biologically plausible system, in both its algorithmic and physical implementation, can also provide insight into tenable biological mechanisms that enable vertebrates to learn to use their bodies while mitigating the risks of injury and overuse—and yet successfully engage in natural selection and predator–prey interactions—which are the Darwinian arbiters in evolutionary success. The ingredients and steps of G2P are all biologically tenable (that is, trial-and-error, memory-based pattern recognition (for example, ref. [51]), Hebbian learning[54,55], experience-based adaptation[42,52]), and allow us to move away from the reasonable, yet arguably anthropocentric and teleological concepts dominating computational neuroscience such as cost functions, optimality, gradients, dimensionality reduction and so on[41–43,46,53]. Although those computational concepts emphasizing optimality are good metaphors, it has been difficult to pin down how one would be able to actually demonstrate their presence and implementation in biological systems[46]. In contrast, G2P can be credibly implementable in biological systems. Our own future investigations aim to demonstrate its implementation as a neuromorphic neuromechanical system, as we have done for other sensorimotor processes[45], as well as developing and modulating the features of more complicated behaviour (such as locomotion) by adding some other hyperparameters to control features, such as step frequency, stride size and so on.

Methods

In this section, we first introduce the control problem by describing the governing dynamics. Next, we go deeper into our learning and control algorithm (software). Finally, we finish this section by providing insight into the physical design of our physical system.

System dynamics. Equation (1) defines the relationship between the joint kinematics and the applied torques of the limb[18] (forward model):

$$\ddot{q} = -I(q)^{-1}C(q,\dot{q}) + B\dot{q} + I(q)^{-1}T \quad (1)$$

where $q \in \mathcal{R}^{2\times1}$, $\dot{q} \in \mathcal{R}^{2\times1}$ and $\ddot{q} \in \mathcal{R}^{2\times1}$ are joint angle vector and its first and second derivatives, respectively, $I \in \mathcal{R}^{2\times2}$ is the inertial matrix, $C(q,\dot{q}) \in \mathcal{R}^{2\times1}$ is the Coriolis and centripetal forces matrix, $B(\dot{q}) \in \mathcal{R}^{2\times2}$ is the joint friction matrix and $T \in \mathcal{R}^{2\times1}$ is the applied joint torque vector. The musculotendon forces (here, cables pulled by the motors) are then related to the applied joint torques vector as described in equation (2):

$$T = M(q)F_0a \quad (2)$$

where $M(q) \in \mathcal{R}^{2\times3}$ is the moment arm matrix, F_0 is a 3×3 diagonal matrix containing the maximal force values that can be exerted by each actuator and $a \in \mathcal{R}^{3\times1}$ is the normalized actuation value of each actuator[42,43]. Please note that this is an under-determined system (three input force values generate two torques) where there is redundancy in the production of net joint torques at each instant. However, because the system is driven by tendons that can pull but not push (and not driven by torque motors coupled directly to the joints, as is common in robotics), joint rotations also depend on the ability of the controller to pay out and reel in tendon as needed, else the movement can be disrupted or the system be non-controllable, respectively (this is why we use back-drivable brushless d.c. motors and maintain tension in the tendons at all times). As such, these tendon-driven systems present the challenge of being simultaneously under- and over-determined[42,43]. The presence of constant tension in the tendons and friction in the joints (which can be heard in Supplementary Video 1) helps stabilize the system but also adds a deadband for the control of subtle movements.

The goal of the inverse map is to find the actuation values vector (a) for any given set of desired kinematics ($q, \dot{q}, \ddot{q}$) without using any implicit model and only from the babbling and task-specific data. The mapping done by the ANN used in the lower-level control of this study is described in equation (3):

$$a = \mathrm{ANN}(q, \dot{q}, \ddot{q}) \quad (3)$$

Finally, the higher-level controller (in the RL task) is in charge of exploring the kinematic space and converging to desired kinematic trajectories that yield high reward. Although these equations are effective for describing and controlling systems, we designed G2P's lower level control with the premise that only the joint dynamics were observable (while not being used in real time), and that the only controllable element is a. As a consequence, our system does not have any direct a priori conception of the model structure or the constants that drive the dynamics; lower level control must infer those relationships using training data from babbling and refine them after each attempt using only task-specific input–output data (without being provided with a desired or error signal while refining the map after each attempt).

Learning and control algorithm. Learning and control in this first implementation of the G2P algorithm takes place at two levels: (1) inverse mapping and refinements (the lower-level control) and (2) the reward-based RL algorithm (the higher-level control). The lower level is responsible for creating an inverse map that converts kinematics into viable control sequences (motor commands). The higher-level control is responsible for reward-driven exploration (RL) of the parametrized kinematics space, which is further passed to the lower-level control and ultimately run through the system.

Inverse mapping and refinements. The lower-level control relies on two phases. As the system is provided with no prior information on its dynamics, topology or structure, it will first explore it dynamics in a general sense by running random control sequences to the motors, which we call motor babbling. After 5 min of motor babbling, the system creates the initial inverse map using the babbling data and then further refines this map using data collected from particular task-specific explorations, which we refer to as task-specific adaptation. This transition from motor babbling to adaptation to a particular task is the reason we refer to this algorithm as 'general to particular' or G2P.

Motor babbling. During this phase, the system tries random control sequences and collects the resulting limb kinematics. A multi-layer perceptron ANN is trained with this input–output set to generate an inverse map between the system inputs (here, motor activation levels) and desired system outputs (here, system kinematics: joint angles, angular velocities and angular accelerations). Although sparse and not tailored for any subsequent task of interest, data from these random inputs and outputs suffice for the ANN to create an approximate general map based on the system's dynamics.

Random activation values for the babbling. The motor activation values (control sequences) for motor babbling were generated using two pseudo-random number generators (uniformly distributed). The first random number generator defines the probability for the activation level to move from one command level to another. This value was set to 1/fs and therefore the activation values for each actuator will change on an average rate of 1 Hz. The second number defines the activation level of the next state with sampling from a range of 15% (to prevent tendons from going slack) to 100% activation. The resulting command signals were stair-step transitions in activations to each motor. Three command signals were created (using different initial random seed), which ran three motors during the motor babbling. It is important to note that these stair-step random activities are designed to explore the general dynamics of the system and are not tailored for any tasks performed during this study (Fig. 6).

Structure of the ANN. The ANN representing the inverse map from 6D limb kinematics to 3D motor control sequences (equation (3)) has three layers (input, hidden and output layers) with 6, 15 and 3 nodes, respectively. The transfer

functions for all nodes were selected as the hyperbolic tangent sigmoid function (with a scaling for the output layer to keep it in the range of the outputs). The performance function was selected as m.s.e. The Levenberg–Marquardt backpropagation technique was used to train the ANN, and weights and biases were initialized according to the Nguyen–Widrow initialization algorithm. Generating and training of ANNs were carried out using MATLAB's Neural Network ToolBox (MathWorks; see MATLAB's Deep Learning Toolbox—formerly known as the Neural Network toolbox—documentation for more details).

Task-based refinements. Motor babbling yields sample observations distributed across a wide range of dynamics, but still represents a sparse sampling of the range of state-dependent dynamic responses of the double pendulum (Fig. 6). As a result, this initial inverse map (ANN_0, Fig. 2) can be further refined when provided with more task-specific data.

The higher-level control will initiate the exploration phase using ANN_0. However, with each exploration, the system is exposed to new, task-specific data, which are appended to the database and incorporated into the refined ANN_K map (Fig. 2). This refinement is achieved by using the current weights as the initial weight of the refined ANN and training it on the cumulative data after each attempt. A validation set is used to stop over-fitting to the training data. The weights will not be updated for a run if the performance over the validation deteriorates for six consecutive attempts (default settings for the used toolbox). The data to be used to train the ANN were randomly divided into train, test and validation sets with 70%, 15% and 15% ratios, respectively. It is important to note that refinements can update the map's validity only up to a point; if major changes to the physical system are experienced (changing the tendon routings or the structure of the system) the network would probably need to retrain on new babbling data. This could be manually performed or a threshold for feedforward error could be set to activate rebabbling. However, we found that motor babbling done strictly while the limb was suspended in the air nevertheless worked well when it was used to produce intermittent contact with the treadmill to produce locomotion on the treadmill and there was no need to rebabble in this study unless a motor, tendon cable or link was replaced.

The reinforcement learning algorithm for the treadmill task. A two-phase reinforcement learning approach is used to systematically explore candidate system dynamics, using a 10D feature vector, ultimately converging to a feature vector that yields high reward. Similar to the ideas used in refs. [56] and [57], we have simplified the search by parametrizing the task as a 10-element feature vector to avoid having the RL agent explore all possible time-varying sequences of motor activations (and their resulting kinematics). We used a 10D feature vector to create cyclical trajectories. The goal of the policy search RL here is to converge to a parameter vector that yields high reward (treadmill movement). The use of a lower-level control to learn the inverse map enabled us to use a policy-based model-free RL with only 10 parameters (feature vector). The system will start from an exploration phase (uniformly random parameter search) and once the reward passes a certain threshold, the policy will change to a multivariate Gaussian distribution-based stochastic search centred on the feature vector that yielded the highest reward so far (see Methods).

Please note that the ANN in the lower-level control only creates an inverse dynamical model between the motor activation values and the joint kinematics (and has no information about the treadmill reward). The RL agent perceives this inverse model simply as a part of the environment. Therefore, this method should not be confused with model-based RL algorithms where the agent utilizes a model to find actions whose predicted reward is maximal.

Creating cyclic trajectories using feature vectors. At each step of the reinforcement algorithm, the policy must produce a candidate set of kinematics. We defined 10 equally distributed spokes (each 36° apart; see Fig. 1c) on the angle–angle space. We can then set the lengths (distance from the centre) of each spoke to define an arbitrary closed path that defines angle changes, which remains a smooth, closed trajectory. The positioning of the spokes and centre are defined by the range of the babbling data. These 10 lengths of the spokes are the 10D feature vector. Using interpolation of these 10 locations, we yield an angle–angle trajectory and derivate those points (equally spaced in the time domain) to obtain the associated angular velocities and accelerations, which fully describe the joint kinematics in the time domain. Using the inverse map (lower-level control) these 6D target limb kinematics $(q, \dot{q}, \ddot{q})$ will be mapped into the associated control sequences. The produced control sequences (motor activation values) are then replicated 20 times and fed to the motors to produce 20 back-to-back repetitions of the cyclical movement. Repeating the task 20 times allows us to smooth the effect of unexpected physical dynamics of the task (for example, system noise, unequal friction values over the treadmill band, nonlinearities of the system and so on), which might lead to fluctuations in reward. The features were bounded in the [0.1–1] range for the treadmill task and [0.2–0.8] during the free cyclical movements experiments to provide more focused task-specific trajectories.

Exploration phase. Exploring random attempts across the 10D feature vector space (uniform at random in [0.1–1]; equation (1)) will eventually produce solutions that yield a treadmill reward. Exploration continues until either the reward is higher than a predefined threshold or stopped when a maximal run number is surpassed (a failure).

Table 1 | Pseudo code for the RL

```
while R < Reward_threshold
   f_bar = Uniform_distribution([0.15, 1]^10)
   R = execute(F_bar)
end
F_best = F_bar
R_best = R
for i = 1:15
   F_bar = Normal_distrubution(F_best, sigma.*Identity(10))
   F_bar = max(min(F_bar, f_M), f_m)
   R = execute(F_bar)
   if R > R_best
      R_best = R
      F_best = F_bar
      sigma = (a-R_best)/b
   end
end
```

Exploitation phase. Once the reward passes the threshold, the system will select a new feature vector in the vicinity of the feature vector from a 10D Gaussian distribution, with each dimension centred at the threshold-jumping solution. Much like a Markov process, with each successful attempt the 10D distribution will be centred on the values of the feature vector that yielded the best reward thus far. The standard deviation of these Gaussian distributions is inversely related to the reward (the distribution will shrink as the system is getting more reward). The minimal standard deviation is bounded at 0.03. This mechanism helps in converging to the behaviour with higher reward and exploring its vicinity in feature space (forming high reward habits) within a reasonable time span but without any guarantee of finding global optima. This is analogous to vertebrate learning behaviour, which can form efficient functional habits that may not be optimal. The governing equations for generating the next feature vector to be executed by the higher-level control are described in equation (4):

$$\overline{F} = \begin{cases} u(f_m, f_M) & R_b < \text{reward threshold} \\ \max(\min(\mathcal{N}(F_b, \sum(R_b)), f_M), f_m) & \text{Otherwise} \end{cases} \quad (4)$$

where u, and $\mathcal{N}$ are Uniform and Gaussian distributions, respectively, $\overline{F}$ is the feature vector of the next attempt, f_m and f_M are the min and max bounds for each feature in the feature vector, respectively (0.1 and 1 in this test), R is the reward, R_b is the highest reward so far, F_b is equal to the feature vector that yielded R_b and $\sum(R_b)$ is described as

$$\sum(R_b) = \sigma(R_b) I_{10} \quad (5)$$

where I_{10} is a 10×10 identity matrix, R is the reward, and sigma is defined as

$$\sigma(R_b) = (b - R_b)/a \quad (6)$$

where a and b are scaling and bias constants, respectively. Here, we empirically selected values of a and b of 600 and 9,000, respectively (Table 1). Note that these values only change the deviation of the feature that will have an impact on the exploration–exploitation trade off; we observed that the performance of the system is not very sensitive to these values (that is, the system will find an acceptable solution as long as reasonable values are set for them).

Between every attempt, the ANN's weights are refined with the accumulated data set (from motor babbling and task-specific trajectories), regardless of the reward or reinforcement phase. This reflects the goal for our system to learn from every experience.

Simulations. We first prototyped our methods in simulation using a double-pendulum model of a tendon-driven limb (equations and code for the double pendulum simulation are adapted, with modifications, from ref. [58]). Similar to the physical system, our method proved to be efficient in the simulation and yielded comparable results (Supplementary Figs. 2 and 3). These simulations were kept

isolated from the physical implementation, and the results were never used as seeds for the physical implementation. It is important to note that, similar to any other modelling attempt, these simulations are simplified representations of the real physics. In addition, some values of the system are very challenging (if not impossible) to measure (for example, the moment arm value function), which is another reason why we think model-free approaches are absolutely necessary in this field. The simulations in this study are mainly designed to test the feasibility of the algorithm before testing it on the real system and are meant to only reflect the general structure of the system, so the parameters of these simulations are not fine-tuned to accurately mimic the physical system.

Physical system. We designed and built a planar robotic tendon-driven limb with two joints (proximal with a fixed height and distal) driven by three tendons, each actuated by a d.c. brushless motor. A passive hinged foot allowed natural contact with the ground. We used d.c. brushless motors as they have low mechanical resistance and are back-drivable. The motor assembly and proximal joint were housed in a carriage that could be lowered or raised to a set elevation for the foot to either reach a treadmill or hang freely in the air (Fig. 3).

We used the minimum number of tendons required to have full control of both joints (a minimum of $n+1$ tendons are required, where n is the number of joints)[42]. Further considerations and part details can be found in the Supplementary Information.

Feasible wrench set and design validation. The feasible force set of a tendon-driven limb is defined by all possible output force vectors it can create. Equation (7) describes the static output wrench for a tendon-driven system[42]:

$$w = J(q)^{-\mathrm{T}} M(q) F_0 a \tag{7}$$

where w represents the wrench (forces and torques) output and $J(q)^{-\mathrm{T}}$ is the Jacobian inverse transpose of the limb, which transforms net joint torques into endpoint wrenches.

By evaluating all binary combinations for the elements in a, the resultant wrenches give rise to a feasible force set. It is important to preserve the physical capability of the tendon routing through the many iterations of limb design, so at each design phase we computed these sets for different positions throughout the limb propulsive stroke. Joint moment arms and tendon routings were simulated and ultimately built to have adequate endpoint torque and forces in all directions, which is important for versatility[42]. Many other effective designs (different tendon routings, different link lengths and so on) or design optimization techniques can be used and their performances in the tasks performed here can be evaluated; however, that is out of the scope of the current study.

Mechanical considerations. The carriage was attached to a wooden support structure via linear bearing and slide rails to adjust its vertical position. A clamp prevented sliding once the vertical position was set. Sandpaper was glued to the footpad and in strips across the treadmill to improve traction (Fig. 3a,b).

Data acquisition. The control system had to provide research-grade accuracy and consistent sampling to enable an effective hardware test of the G2P. A Raspberry Pi (Raspberry Pi Foundation) served as a dedicated control loop operator—issuing commands to the motors, sensing angles at each of the proximal and distal joints, and recording the treadmill band displacement (Fig. 3a,b). Furthermore, the electrical power consumption for each motor was measured at 500 Hz using current-sensing resistors in parallel with the motor drivers, calculating the watt-hours over each intersample interval and reporting the amortized mean power (watts) for the entire attempt. All commands were sent, and data received, via WiFi communication with the Raspberry Pi as csv files.

Running the system. The limb was placed in a consistent starting posture before activations were run to minimize variance in the initial conditions of the physical system. To aid development, a live-streaming video feed was designed for real-time visualization on any computer on the network (Supplementary Video 1). A computer sent a control sequence to the Raspberry Pi, and after it was successfully run, the computer received (1) the paired input-to-output data in csv format for iterative analysis or training, (2) the net distance (mm) covered over the course of the entire action and (3) the amortized power the system consumed during the trial. Once the data had been collected, to calculate kinematics to train the inverse map, samples were first interpolated using their corresponding time labels to combat the non-uniform intersample interval of 78 ± 5 Hz. Prescribed activation trajectories were also served at this rate. The pipeline for data acquisition was designed with Python 3.6.

Data availability

The source code can be accessed at https://github.com/marjanin/Marjaninejad-et.-al.—2019-NMI.

All other data (run data for experiments as well as the 3D printing files) can be accessed at https://drive.google.com/drive/folders/1FO0QJ2fBsdYCJs-h1LH7Iwb-wa0VPDi-?usp=sharing

Received: 22 September 2018; Accepted: 5 February 2019;
Published online: 11 March 2019

References

1. Lowrey, K., Kolev, S., Dao, J., Rajeswaran, A. & Todorov, E. Reinforcement learning for non-prehensile manipulation: transfer from simulation to physical system. In *Proc. 2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)* 35–42 (IEEE, 2018).
2. Andrychowicz, M. et al. Learning dexterous in-hand manipulation. Preprint at https://arxiv.org/abs/1808.00177 (2018).
3. Kobayashi, H. & Ozawa, R. Adaptive neural network control of tendon-driven mechanisms with elastic tendons. *Automatica* **39**, 1509–1519 (2003).
4. Nguyen-Tuong, D., Peters, J., Seeger, M. & Schölkopf, B. Learning inverse dynamics: a comparison. In *Proc. European Symposium on Artificial Neural Networks* 13–18 (2008).
5. Osa, T., Peters, J. & Neumann, G. Hierarchical reinforcement learning of multiple grasping strategies with human instructions. *Adv. Robot.* **32**, 955–968 (2018).
6. Manoonpong, P., Geng, T., Kulvicius, T., Porr, B. & Wörgötter, F. Adaptive, fast walking in a biped robot under neuronal control and learning. *PLoS Comput. Biol.* **3**, e134 (2007).
7. Marques, H. G., Bharadwaj, A. & Iida, F. From spontaneous motor activity to coordinated behaviour: a developmental model. *PLoS Comput. Biol.* **10**, e1003653 (2014).
8. Gijsberts, A. & Metta, G. Real-time model learning using incremental sparse spectrum Gaussian process regression. *Neural Netw.* **41**, 59–69 (2013).
9. Della Santina, C., Lakatos, D., Bicchi, A. & Albu-Schäffer, A. Using nonlinear normal modes for execution of efficient cyclic motions in soft robots. Preprint at https://arxiv.org/abs/1806.08389 (2018).
10. Bongard, J., Zykov, V. & Lipson, H. Resilient machines through continuous self-modeling. *Science* **314**, 1118–1121 (2006).
11. Krishnan, S. et al. SWIRL: A sequential windowed inverse reinforcement learning algorithm for robot tasks with delayed rewards. *Int. J. Rob. Res.* https://doi.org/10.1177/0278364918784350 (2018).
12. James, S. et al. Sim-to-Real via Sim-to-Sim: Data-efficient Robotic Grasping via Randomized-to-Canonical AdaptationNetworks. Preprint at https://arxiv.org/abs/1812.07252 (2018).
13. Takahashi, K., Ogata, T., Nakanishi, J., Cheng, G. & Sugano, S. Dynamic motion learning for multi-DOF flexible-joint robots using active–passive motor babbling through deep learning. *Adv. Robot.* **31**, 1002–1015 (2017).
14. Marco, A., Hennig, P., Bohg, J., Schaal, S. & Trimpe, S. Automatic LQR tuning based on Gaussian process global optimization. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* 270–277 (IEEE, 2016).
15. Geijtenbeek, T., Van De Panne, M. & Van Der Stappen, A. F. Flexible muscle-based locomotion for bipedal creatures. *ACM Trans. Graph.* **32**, 206 (2013).
16. Kumar, V., Tassa, Y., Erez, T. & Todorov, E. Real-time behaviour synthesis for dynamic hand-manipulation. In *Proc. 2014 IEEE International Conference on Robotics and Automation (ICRA)* 6808–6815 (IEEE, 2014).
17. Kumar, V., Gupta, A., Todorov, E. & Levine, S. Learning dexterous manipulation policies from experience and imitation. Preprint at https://arxiv.org/abs/1611.05095 (2016).
18. Rombokas, E., Theodorou, E., Malhotra, M., Todorov, E. & Matsuoka, Y. Tendon-driven control of biomechanical and robotic systems: a path integral reinforcement learning approach. In *Proc. 2012 IEEE International Conference on Robotics and Automation (ICRA)* 208–214 (IEEE, 2012).
19. Potkonjak, V., Svetozarevic, B., Jovanovic, K. & Holland, O. The puller–follower control of compliant and noncompliant antagonistic tendon drives in robotic systems. *Int. J. Adv. Robot. Syst.* **8**, 69 (2011).
20. Hunt, A., Szczecinski, N. & Quinn, R. Development and training of a neural controller for hind leg walking in a dog robot. *Front. Neurorobot.* **11**, 18 (2017).
21. Fazeli, N. et al. See, feel, act: hierarchical learning for complex manipulation skills with multisensory fusion. *Sci. Robot.* **4**, eaav3123 (2019).
22. Rasmussen, D., Voelker, A. & Eliasmith, C. A neural model of hierarchical reinforcement learning. *PLoS One* **12**, e0180234 (2017).
23. Parisi, S., Ramstedt, S. & Peters, J. Goal-driven dimensionality reduction for reinforcement learning. In *Proc. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 4634–4639 (IEEE, 2017).
24. D'Souza, A., Vijayakumar, S. & Schaal, S. Learning inverse kinematics. *Intell. Robots Syst.* **1**, 298–303 (2001).
25. Bonarini, A., Lazaric, A. & Restelli, M. Incremental skill acquisition for self-motivated learning animats. In *Proc. International Conference on Simulation of Adaptive Behavior* 357–368 (Springer, 2006).
26. Najjar, T. & Hasegawa, O. Self-organizing incremental neural network (SOINN) as a mechanism for motor babbling and sensory-motor learning in

developmental robotics. In *Proc. International Conference on Artificial Neural Networks* 321–330 (Springer, 2013).
27. Marjaninejad, A., Annigeri, R. & Valero-Cuevas, F. J. Model-free control of movement in a tendon-driven limb via a modified genetic algorithm. In *40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (IEEE, 2018).
28. Rajeswaran, A. et al. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. Preprint at https://arxiv.org/abs/1709.10087 (2017).
29. Schulman, J., Levine, S., Abbeel, P., Jordan, M. & Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning* 1889–1897 (PMLR, 2015).
30. Mnih, V. et al. Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015).
31. Salimans, T., Ho, J., Chen, X., Sidor, S. & Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning. Preprint at https://arxiv.org/abs/1703.03864 (2017).
32. Vinyals, O. et al. Starcraft II: a new challenge for reinforcement learning. Preprint at https://arxiv.org/abs/1708.04782 (2017).
33. Metta, G. et al. The iCub humanoid robot: an open-systems platform for research in cognitive development. *Neural Netw.* **23**, 1125–1134 (2010).
34. Pathak, D., Agrawal, P., Efros, A. A. & Darrell, T. Curiosity-driven exploration by self-supervised prediction.Iin *International Conference on Machine Learning (ICML) 2017*, (2017).
35. Luo, Q. et al. Design of a biomimetic control system for tendon-driven prosthetic hand. In *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)* 528–531 (2018).
36. Ravi, S. & Larochelle, H. Optimization as a model for few-shot learning. In *Proc. ICLR* (2016).
37. Schaal, S. in *Humanoid Robotics: A Reference*. (eds Goswami, A. & Vadakkepat, P.) 1–9 (Springer, Dordrecht, 2018).
38. Bohg, J. et al. Interactive perception: leveraging action in perception and perception in action. *IEEE Trans. Robot.* **33**, 1273–1291 (2017).
39. Ingram, T. G. J., Solomon, J. P., Westwood, D. A. & Boe, S. G. Movement related sensory feedback is not necessary for learning to execute a motor skill. *Behav. Brain Res.* **359**, 135–142 (2019).
40. Fine, M. S. & Thoroughman, K. A. Trial-by-trial transformation of error into sensorimotor adaptation changes with environmental dynamics. *J. Neurophysiol.* **98**, 1392–1404 (2007).
41. Adolph, K. E. et al. How do you learn to walk? Thousands of steps and dozens of falls per day. *Psychol. Sci.* **23**, 1387–1394 (2012).
42. Valero-Cuevas, F. J. *Fundamentals of Neuromechanics* 8 (Springer, New York, NY, 2015).
43. Marjaninejad, A. & Valero-Cuevas, F. J. in *Biomechanics of Anthropomorphic Systems* (eds Venture, G., Laumond, J.-P. & Watier, B.) 7–34 (Springer, New York, NY, 2019).
44. McAndrew, P. M., Wilken, J. M. & Dingwell, J. B. Dynamic stability of human walking in visually and mechanically destabilizing environments. *J. Biomech.* **44**, 644–649 (2011).
45. Jalaleddini, K. et al. Neuromorphic meets neuromechanics. Part II: The role of fusimotor drive. *J. Neural Eng.* **14**, 025002 (2017).
46. Loeb, G. E. Optimal isn't good enough. *Biol. Cybern.* **106**, 757–765 (2012).
47. Collins, S. H., Wiggin, M. B. & Sawicki, G. S. Reducing the energy cost of human walking using an unpowered exoskeleton. *Nature* **522**, 212–215 (2015).
48. Kobayashi, T., Sekiyama, K., Hasegawa, Y., Aoyama, T. & Fukuda, T. Unified bipedal gait for autonomous transition between walking and running in pursuit of energy minimization. *Rob. Auton. Syst.* **103**, 27–41 (2018).
49. Finley, J. M. & Bastian, A. J. Associations between foot placement asymmetries and metabolic cost of transport in hemiparetic gait. *Neurorehabil. Neural Repair* **31**, 168–177 (2017).
50. Selinger, J. C., O'Connor, S. M., Wong, J. D. & Donelan, J. M. Humans can continuously optimize energetic cost during walking. *Curr. Biol.* **25**, 2452–2456 (2015).
51. Zhang, W., Gordon, A. M., Fu, Q. & Santello, M. Manipulation after object rotation reveals independent sensorimotor memory representations of digit positions and forces. *J. Neurophysiol.* **103**, 2953–2964 (2010).
52. Wolpert, D. M. & Flanagan, J. R. Computations underlying sensorimotor learning. *Curr. Opin. Neurobiol.* **37**, 7–11 (2016).
53. Todorov, E. Optimality principles in sensorimotor control. *Nat. Neurosci.* **7**, 907–915 (2004).
54. Grillner, S. Biological pattern generation: the cellular and computational logic of networks in motion. *Neuron* **52**, 751–766 (2006).
55. Hebb, D. O. *The Organization of Behavior: A Neuropsychological Theory* (Wiley, New York, NY, 1949).
56. Ijspeert, A. J., Nakanishi, J. & Schaal, S. in *Advances in Neural Information Processing Systems* Vol. 15 (eds Becker, S., Thrun, S. & Obermayer, K.) 1547–1554 (MIT Press, Cambridge, MA, 2003).
57. Feirstein, D. S., Koryakovskiy, I., Kober, J. & Vallery, H. Reinforcement learning of potential fields to achieve limit-cycle walking. In *Proc. 6th IFAC Workshop on Periodic Control System* Vol. 49, 113–118 (Elsevier, 2016).
58. http://ruina.tam.cornell.edu/research/topics/locomotion_and_robotics/ranger/ranger_paper/Reports/Ranger_Robot/control/simulator/doublependulum.html

Acknowledgements

The authors thank H. Zhao for support in designing and manufacturing the physical system as well as support in the analysis of the limb kinematics, S. Kamalakkannan for support in designing and implementing the data acquisition system, and Y. Kahsai for Figs. 1 and 2. Research reported in this publication was supported by the National Institute of Arthritis and Musculoskeletal and Skin Diseases of the National Institutes of Health under award numbers R01 AR-050520 and R01 AR-052345 to F.J.V.-C. This work was also supported by Department of Defense CDMRP Grant MR150091 and Award W911NF1820264 from the DARPA-L2M programme to F.J.V.-C. The authors acknowledge additional support for A.M. for Provost and Research Enhancement Fellowships from the Graduate School of the University of Southern California and fellowships for D.U.-M. from the Consejo Nacional de Ciencia y Tecnología (Mexico) and for B.C. from the NSF Graduate Research Fellowship Program. The content of this endeavour is solely the responsibility of the authors and does not represent the official views of the National Institutes of Health, the Department of Defense, The National Science Foundation nor the Consejo Nacional de Ciencia y Tecnología.

Author contributions

All authors contributed to the conception and design of the work and writing of the manuscript. A.M. led the development of the G2P algorithm, D.U.-M. led the construction of the robotic limb and B.A.C. led the data acquisition and analysis. F.J.V.-C. provided general direction for the project. All authors approved the final version of the manuscript and agree to be accountable for all aspects of the work. All persons designated as authors qualify for authorship, and all those who qualify for authorship are listed.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s42256-019-0029-0.

Reprints and permissions information is available at www.nature.com/reprints.

Correspondence and requests for materials should be addressed to F.J.V.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© The Author(s), under exclusive licence to Springer Nature Limited 2019