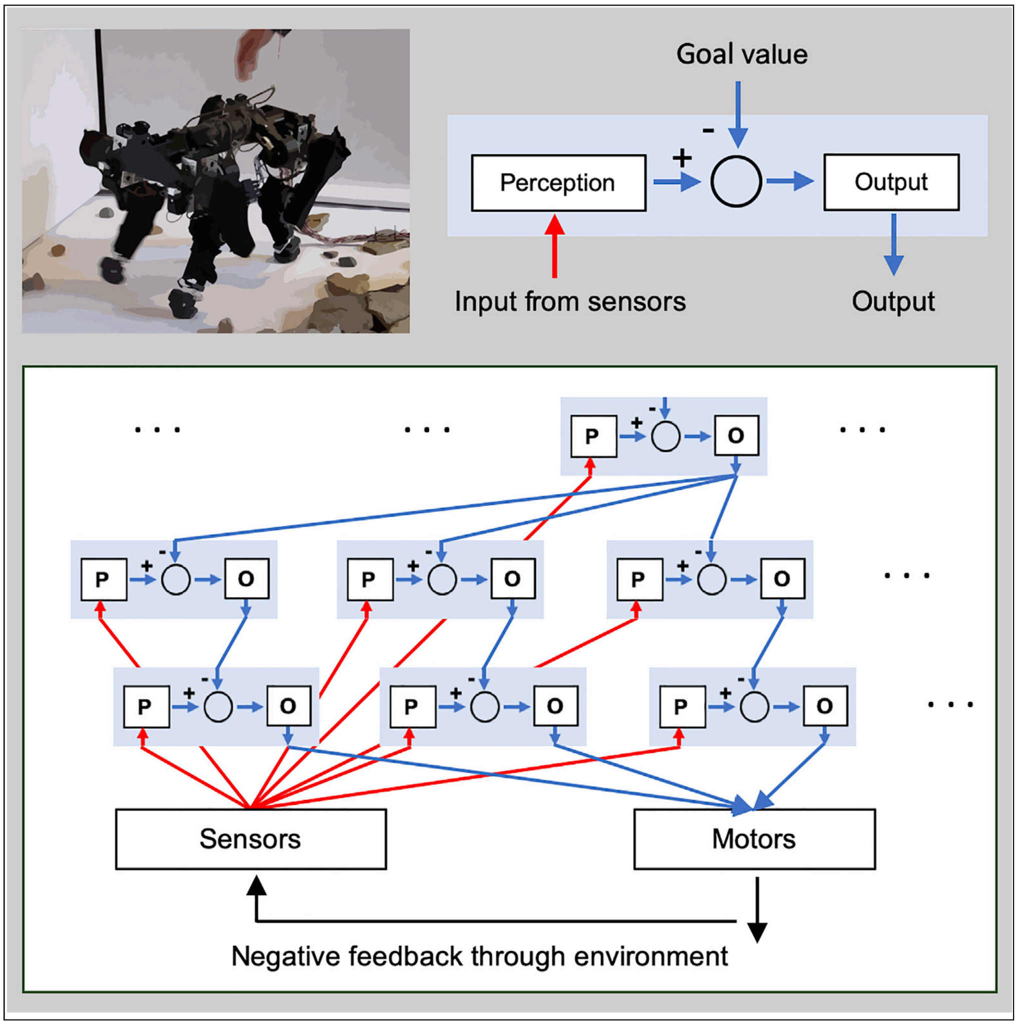
Article

Achieving natural behavior in a robot using neurally inspired hierarchical perceptual control



Joseph W. Barter,
Henry H. Yin

hy43@duke.edu

Highlights

Inspired by a neural hierarchy with control of input at each level

Higher level specifies reference state for lower level

Successful posture control and locomotion despite unpredictable disturbances

No need for training or computation of inverse or forward kinematics

Barter & Yin, iScience *24*, 102948
September 24, 2021 © 2021 The Authors.
https://doi.org/10.1016/j.isci.2021.102948

Article

Achieving natural behavior in a robot using neurally inspired hierarchical perceptual control

Joseph W. Barter[1] and Henry H. Yin[1,2,3,*]

SUMMARY

Terrestrial locomotion presents tremendous computational challenges on account of the enormous degrees of freedom in legged animals, and complex, unpredictable properties of natural environments, including the body and its effectors, yet the nervous system can achieve locomotion with ease. Here we introduce a quadrupedal robot that is capable of posture control and goal-directed locomotion across uneven terrain. The control architecture is a hierarchical network of simple negative feedback control systems inspired by the organization of the vertebrate nervous system. This robot is capable of robust posture control and locomotion in novel environments with unpredictable disturbances. Unlike current robots, our robot does not use internal inverse and forward models, nor does it require any training in order to perform successfully in novel environments.

INTRODUCTION

The generation of adaptive locomotion is arguably the most fundamental function of the nervous system, but the underlying mechanisms remain poorly understood, especially for legged vertebrates with complex multijointed bodies (Grillner, 2011). Movement cannot simply be equated with muscle output. Rather, muscle outputs must vary by the correct amount to counter precisely the effects of continuously varying and unpredictable disturbances in any natural environment (Bernstein, 1967; Yin, 2013). But this observation also presents a major challenge: In any forward computation of the required neural output to successfully produce behavior, it would be necessary to take into account imperfect sensors, changes in muscle properties over time, a high level of redundancy in effectors, and unpredictable environmental conditions. This problem is made exponentially harder with increasing degrees of freedom (Wolpert and Ghahramani, 2000).

There is an alternative approach that does not require forward computation. It assumes that movements are achieved by continuously controlling *perceptual inputs* through negative feedback (Powers, 1973; Powers et al., 1960). Sensory perceptions are simply compared with internal goals and the errors between them drive corrective outputs that influence those perceptions via a feedback function. For such control systems to deal with the challenges of the natural environment, however, a particular hierarchical organization is required. In such a hierarchy, higher level controllers achieve their own reference states by adjusting the reference states of lower systems. Only the lowest systems have access to the effectors that act on the environment. For example, sensed muscle tension and length are controlled at lower levels in the spinal cord, whereas progressively more abstract variables including body posture, orientation, and position in space are controlled at higher levels in the brainstem, midbrain, and basal ganglia (Yin, 2014, 2017).

To test whether this hierarchical model can successfully generate natural behavior, we developed a quadrupedal robot with a hierarchical perceptual control architecture. This robot is capable of generating posture control, locomotion, orientation, and basic navigation across uneven and unstable terrain.

RESULTS

The robot is a quadruped with 12 degrees of freedom in four three-jointed legs (Figure 1A). Each joint is equipped with an analog torque sensor and an analog joint angle sensor. Each foot has a ground reaction force sensor, and the trunk houses an inertial measurement unit (IMU) for sensing tilts of the body (Figure 1B).

[1]Department of Psychology and Neuroscience, Duke University, Durham, NC 27708, USA

[2]Department of Neurobiology, Duke University School of Medicine, Durham, NC 27708, USA

[3]Lead contact

*Correspondence: hy43@duke.edu

https://doi.org/10.1016/j.isci.2021.102948

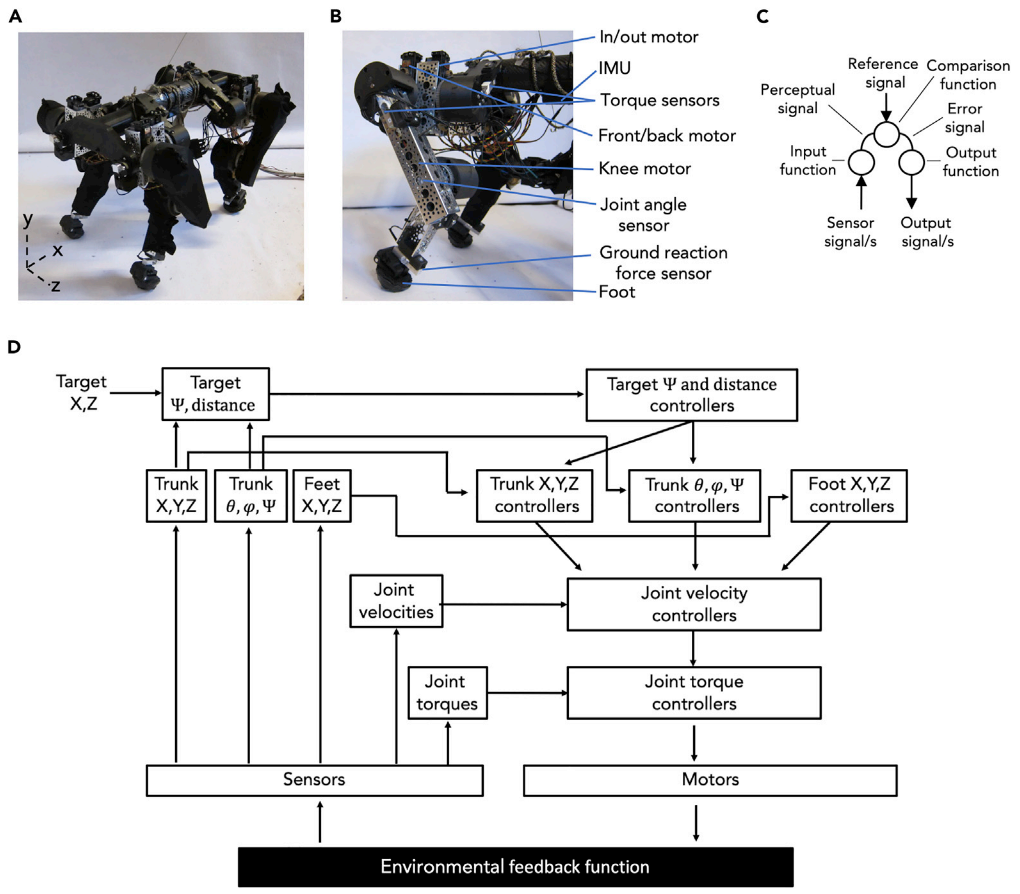


Figure 1. System overview

(A) Image of robot standing with neutral posture.

(B) Skin cutaway of a single leg showing leg parts. For each leg there are three joint angle sensors and three torque sensors, one on each joint. Also shown is the IMU for the body.

(C) Top: A simplified schematic of the closed loop control system module: a signal or signals are read into the input function where they are converted into a single perceptual value, which is compared with an internal reference value to generate an error. The error is converted by the output function into an output signal, which is sent out to a lower system.

(D) Block diagram showing signal flow in our control architecture. Sensor inputs are converted by perceptual functions into joint torque perceptions; joint angular velocity perceptions; trunk X, Y, Z position and pitch (θ); roll (ϕ) and yaw (Ψ) perceptions. Yaw orientation to target is determined from target X, Z position and the trunk yaw perception. Distance to target is determined from target X, Z position and trunk X, Z position. Target position here is set by the experimenter. The motor output operates through the environment to influence sensory input.

All control systems at every level of the hierarchy have the same basic design. They are distinguished by the type of the perceptual signal that each system receives and controls; and they vary also in their output tuning, output conversion functions, and connectivity with other controllers. Each controller has three basic parts (Figure 1C): (1) an input function that converts raw sensory input into a single perceptual value; (2) a comparison function that compares this perceptual value with a reference value for that perception and computes the difference between them; and (3) an output function that transforms the error into output using proportional-derivative (PD) tuning, and the resultant output will become reference signals for the appropriate set of lower level control systems.

Sensory inputs are read, and outputs are commanded, in a continuous loop operating at a frequency of ~230 Hz (Figure 1D). During each iteration of the loop and starting at the top of the hierarchy, the perceptual value for each control system module is compared with a corresponding reference value to compute an error signal. The error from each active control system is then converted into the appropriate set of reference values for controllers at the level immediately below. At the lowest level, errors are converted into output commands for the motors.

Note that, in all controllers, the variable being controlled is always the perceptual variable from the input function and *not* the output, as often described in control engineering. The output of any module is not the result of sensorimotor transformation but varies as a function of the difference between current internal goals and perceptual inputs. The highest-level controllers at the fourth level are commanded by a sequence of target positions specified by the experimenter.

Network levels 1–2: joint controllers

The lowest level in this control hierarchy performs joint torque control. For each joint in the body, torque is sensed and controlled by a dedicated controller. As with muscle tension control in a real animal, torque controllers may have a "ceiling"; torque inputs beyond these safe limits will cause the joint to yield, protecting it from damage. For the purposes of locomotor behavior, joint torque perceptions are individually normalized so that a value of zero corresponds to the baseline value measured while standing.

At the second level of the hierarchy, joint angular velocity is controlled. For each joint, angular velocity is sensed and compared with the reference angular velocity. The angular velocity error is converted into a reference signal for the corresponding joint torque controller at level 1. The angular velocity error and torque reference are both reduced through movement commanded by the torque controller.

This velocity-torque hierarchy enables a form of viscous damped compliance. With a fixed angular velocity reference signal, any outside forces acting upon the joint will cause it to yield with a resistance roughly proportional to the velocity of the disturbance, as the angular velocity error is converted into a torque reference signal to oppose that error. Thus, the angular velocity controller gain determines the amount of damping. This feature, which is partly inspired by the lowest spinal circuits for controlling muscle tension and length, provides stabilizing and protective benefits of damped compliance.

Network levels 3–4: posture and locomotor controllers

At the third level, the position and orientation of body parts are perceived and controlled. Because the lengths of body segments are known, we can convert a vector of joint angle measurements to body position in 3D space using trigonometry. With an IMU to sense the trunk tilt angle, it is possible through a reference frame rotation to obtain body position relative to gravity.

Foot ground reaction force sensors are used to detect ground contact. As in the lower controllers, controllers at this level operate by computing the difference between reference and perception. Errors are then converted into the appropriate set of reference signals for joint angular velocity controllers. A single error can be used to generate a set of reference signals for a group of lower controllers (Figures 2A and 2B).

Because the conversion between a given movement in Cartesian space and the corresponding joint angles is non-linear and depends on the current body posture, movements can be stabilized by including a simple gain scheduler in the output function that takes the current body posture into account. This gain scheduler preserves a linear relationship between the error and output regardless of the current body posture. With fixed gains, movements could be too large in certain postures and too small in others. For example, trunk height control in crouched versus tiptoe postures require different joint angle changes to accomplish the same amount of vertical movement. The schedulers dynamically adjust the effective gain of each output signal from a level 3 controller to lower joint angular velocity controllers in order to scale the output appropriately. The gain scheduler is a part of the output function of each level 3 postural controller.

Two types of perceptual variables are controlled at level 3: the positioning of each foot and the positioning of the trunk. These controllers are antagonistic: the trunk controllers only operate through a given leg when the foot is planted on the ground, whereas the foot controllers only operate through that leg if the foot is unweighted and free to move about (Figures 2A and 2B). For each leg there is one foot position controller for each of the 3 Cartesian axes of movement relative to gravity. The y axis controller perceives the position of the foot relative to the previous ground surface, while the x and z axis controllers perceive the position of the foot relative to its position during neutral standing posture. Each foot position controller works by

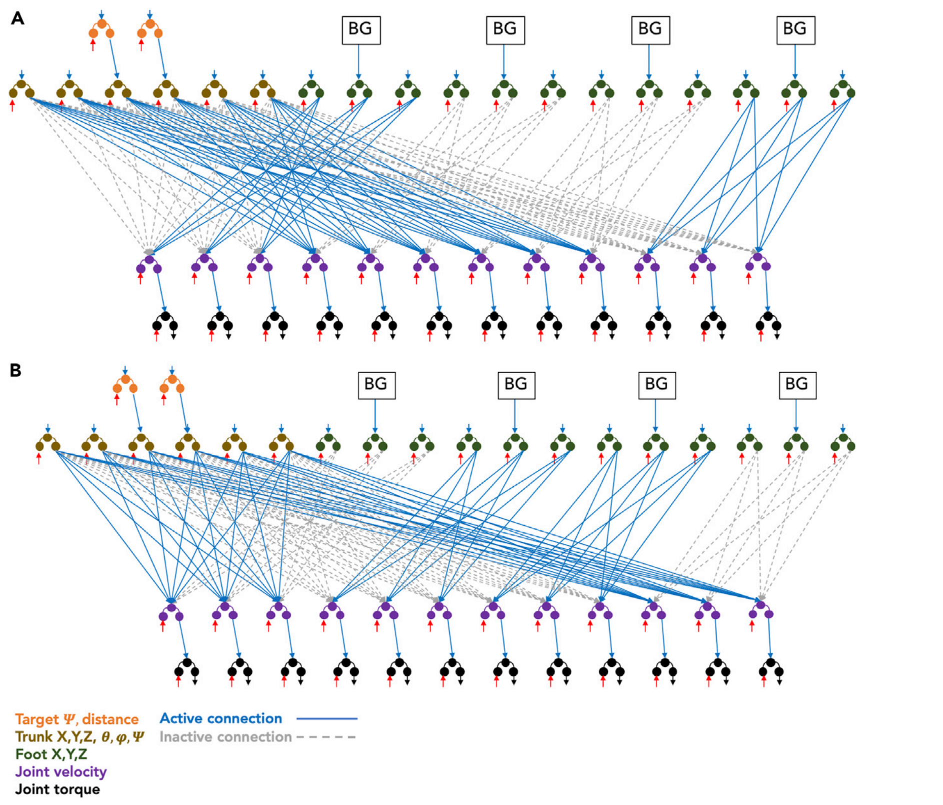


Figure 2. Network architecture diagram illustrating reference signal connections and antagonism during stepping

(A and B) Top and bottom diagrams both show the same network architecture. Individual control system modules are represented using the same schematic format as in Figure 1C; each system receives an input signal and a reference signal and outputs a perceptual signal. Blue lines represent active reference signals, red arrows represent perceptual inputs, and dashed gray lines represent inactive reference signals. The hierarchy architecture is defined by how output signals of higher systems become reference signals of lower systems. The top (A) diagram shows active and inactive network connections during locomotion when the FL and BR feet are up and FR,BL feet are down. The joint velocity reference signals of the down legs are determined by the trunk orientation and position controllers, while the joint velocity reference signals of the up legs are determined by the foot position controllers. Orange controllers at level 4 represent target yaw (Ψ) orientation and position controllers. Brown controllers at level 3 represent trunk pitch (θ), roll (ϕ), and yaw (Ψ) orientation and X,Y,Z position controllers. Green controllers at level 3 represent foot X,Y,Z position controllers. Purple controllers at level 2 represent joint angle velocity controllers. Black controllers at level 1 represent joint torque controllers. The boxes labeled "BG" represent the burst generators. There is one burst generator for each leg. Burst generators activate the foot position controllers and adjust the Y (foot height) reference signal. The bottom (B) diagram shows the same thing as in (A) except with the leg pairs reversed.

converting its error into three reference signals: one for each joint velocity controller of the same leg (Figure 3A). When the X, Y, and Z controllers for a single foot are running with fixed reference signals, then the foot remains fixed in space despite unpredictable trunk movements (Figure 3B). Voluntary movement of the foot by changing the position reference signals is also stable (Figure 3C), even when the body is being tipped unpredictably (Figure 3D). This capacity is important for preventing feet from jamming into the ground during locomotion when the trunk is disturbed.

For trunk orientation, there are 3 controllers: pitch, roll, and yaw. Pitch and roll are perceived directly from the IMU, whereas yaw is perceived through integration of trunk yaw velocity relative to the feet that are planted on the ground. Each trunk orientation controller operates through the legs that are planted on the ground. If 4 legs are planted, then the controller will adjust the reference signals of 12 lower joint velocity controllers; if 2 legs are planted, then the controller will adjust only 6 controllers (Figure 4A). Trunk orientation control can maintain its reference state when the ground is being unpredictably tipped

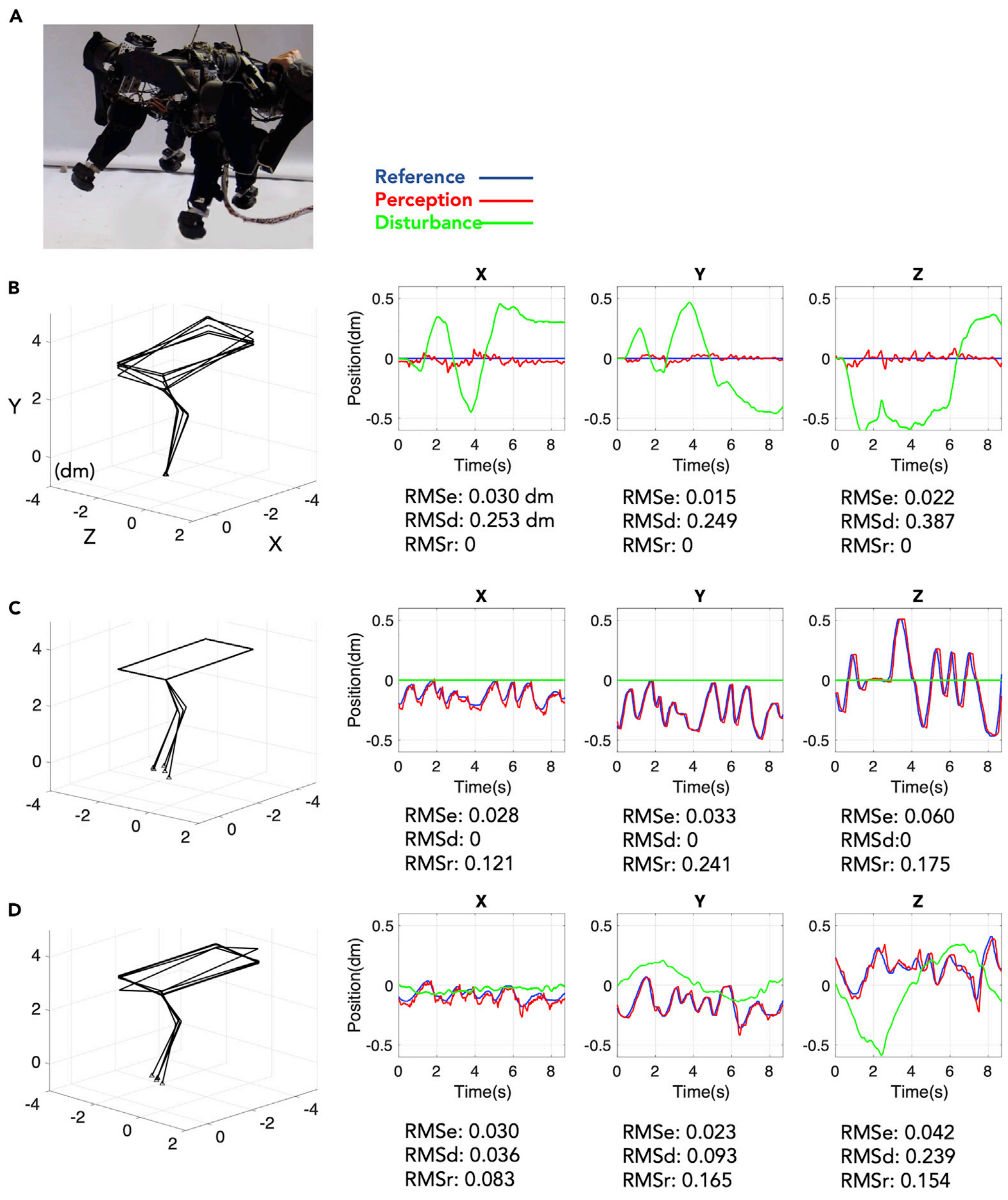


Figure 3. Foot position control

(A) The robot is shown hanging so that the trunk can be tipped while the legs are free to move.

(B) Representative plots of X, Y, and Z position control in the front left (FL) foot of a hanging robot with fixed X, Y, and Z position reference signals while the experimenter randomly tips the body. Red lines represent perceptual signal, blue lines represent reference signal, and green lines represent the disturbance, measured as the position of the foot due to body tilt if the controllers were off and the leg were rigid. For each plot, root mean squared error (RMSe), root mean squared disturbance (RMSd), and root mean squared reference signal (RMSr) values are reported, showing that the error is held small despite large changes in the disturbance. Wireframe snapshots illustrate foot stability, matching the stable reference signal, despite disturbances causing body movement.

(C) Same plot format as in (B), showing a scenario in which the robot is not tipped but the foot position reference signals are randomly changed by the experimenter. RMSe is held at a low value despite large changes in RMSr.

(D) Same plot format as above, showing a scenario in which the trunk is randomly tipped while the foot position reference signals are also randomly changed. RMSe is held at a low value despite large changes in both RMSd and RMSr.

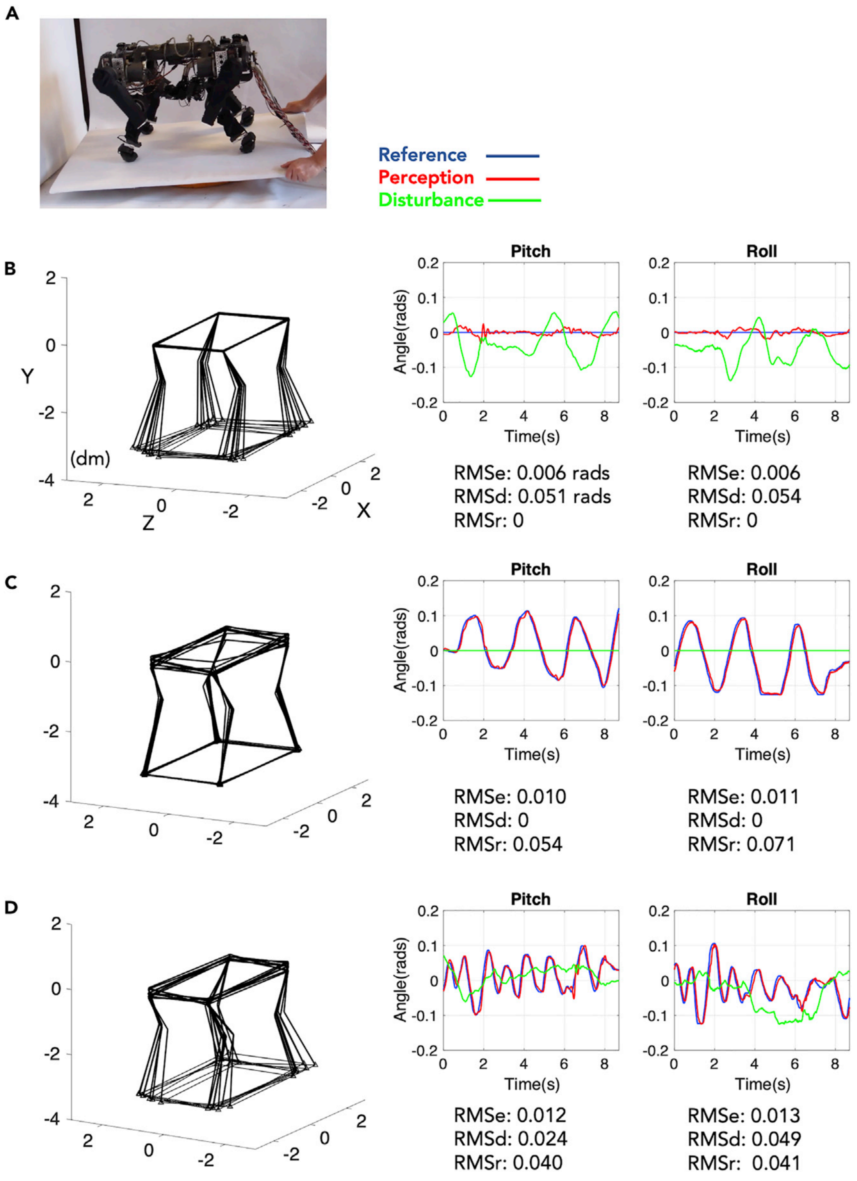


Figure 4. Trunk orientation control

(A) The robot is shown standing on a tippable platform.

(B) Representative plots of trunk pitch and roll control while the experimenter randomly tips the platform. Red lines represent perceptual signal, blue lines represent reference signal, and green lines represent the disturbance, measured as the angle of the platform. For each plot, RMSe, RMSd, and RMSr values are reported, showing that the error is held small despite large changes in the disturbance.

(C) Same plot format as in (B), showing a scenario in which the robot is not tipped but the pitch and roll reference signals are randomly changed by the experimenter. RMSe is held at a low value despite large changes in RMSr.

(D) Same plot format as above, showing a scenario in which the platform is randomly tipped while the pitch and roll reference signals are also randomly changed. RMSe is held at a low value despite large changes in both RMSd and RMSr.

(Figure 4B; Video S1), when the ground is unmovingwhile the trunk is being voluntarily tipped through reference signal change (Figure 4C; Video S2), or when the ground is being tipped while the trunk is being voluntarily tipped at the same time (Figure 4D). During locomotion, pitch and roll reference signals were left at neutral values. Trunk pitch may also be perceived relative to the pitch of the ground surface as sensed by the feet so that the trunk remains parallel to the ground. We have found this mode to be helpful in allowing the legs to maintain a neutral posture during locomotion across uneven terrain.

For trunk position control at level 3, there are three controllers: X, Y, and Z. Trunk Y height is perceived by comparing the trunk position along that axis to the average position of the feet that are planted on the ground. X and Z are perceived as the path-integrated position of the trunk relative to the legs that are and have been planted on the ground. As with the trunk orientation controllers, these trunk position controllers operate through whichever legs are on the ground, so that the trunk can move while the feet are stationary. Each down leg expresses the added output of all 6 trunk controllers. Video S3 shows control of trunk height and yaw while the reference signals are arbitrarily changed by the experimenter.

Rhythmic stepping is accomplished by a burst generator for each leg. When activated, a burst generator will switch the leg from trunk control to foot position control at the same time as it adjusts the Y foot position reference signal so that the foot lifts up and then back down towards the ground. When the ground reaction force sensor senses contact with the ground after achieving lift height, the burst generator resets and the leg switches back to effecting trunk control (Figure 5A). Once both burst generators in a pair are deactivated following a burst, then the opposite pair of burst generators is activated. Although simplified, this design is similar to published models of central pattern generators (CPGs) for locomotion, in which burst generators are connected with reciprocal innervation to form coupled oscillators (Grillner et al., 2008; McCrea and Rybak, 2007, 2008). As in these classic models, the CPGs in our robot are modulated by sensory feedback. For example, ground contact terminates a burst. However, this approach is different from previous approaches in that our CPGs operate within the framework of closed loop hierarchical control. They only activate the foot position controllers and adjust the reference signal to cause stepping along the vertical axis. Foot positioning itself and target pursuit, trunk movement, balance, and postural control are all accomplished by the control hierarchy. Although CPGs are important components of locomotor control, by themselves they are not sufficient for natural locomotion as they lack the feedback needed to defend posture, balance, and relationship to environmental targets against unpredicted disturbances. The primary challenge in locomotion is not pattern generation, but control. In principle, pattern generation can be achieved in many different ways, but the type of control required in locomotion appears to require the hierarchical control organization described here.

Because of motor speed limitations, step height achieved during locomotion is typically below 2 cm. Burst generators are activated in diagonal pairs so that the termination of both burst generators of a diagonal pair activates those of the opposite pair. Using this scheme, simply changing the reference signals for yaw orientation (Figure 5B; Video S4) and trunk X position (Video S5) during stepping result in locomotion. Trunk Y and Z position reference signals stay at a neutral zero value. When in swing mode, a foot will reach for a new X, Z reference position that reflects the velocity of the trunk, so that the feet step in the direction of trunk movement, whether during voluntary locomotion or in response to an environmental disturbance (Video S6).

At the fourth level in the hierarchy, orientation and distance to any specific location in the environment are controlled through locomotion. Since the robot is blind and has no distal sensors, position and orientation in the world are perceived by continuously integrating yaw velocity and translation velocity of the trunk by the down legs. Orientation relative to an external target is controlled by converting error into trunk yaw reference signal at level 3, whereas distance to the target is controlled by converting error into X (front-back) trunk position reference. In level 4 controllers of this robot, the reference signals are left at a neutral zero value so that the robot always orients toward a target and seeks to reduce the target distance to zero. In these controllers, the target location is defined as part of the perceptual function and target locations are given by the experimenter.

Level 4 outputs are integrated so that they are converted into gradual changes in level 3 reference signals, allowing gradual changes in actual position and orientation. Robot trunk velocity is proportional to level 4 outputs. As shown in Figure 5C, by controlling trunk position and orientation relative to targets in the world, the robot is able to navigate a path through the environment according to a sequence of target position references specified by the experimenter. The robot is able to successfully reach each target in a 3-step sequence over flat terrain (Video S7) and over uneven and unstable terrains, including loose rocks (Video S8) and rock piles (Video S9; Figure 5C). Figure 6A shows several different terrains and corresponding target approach signals. The trunk controllers not only generate movement but also defend against disturbances from external sources such as an experimenter push (Figure 6B), as well as the continuous disturbances from stepping itself (Figure 6C-D). A summary of robot performance is shown in Figure 7.

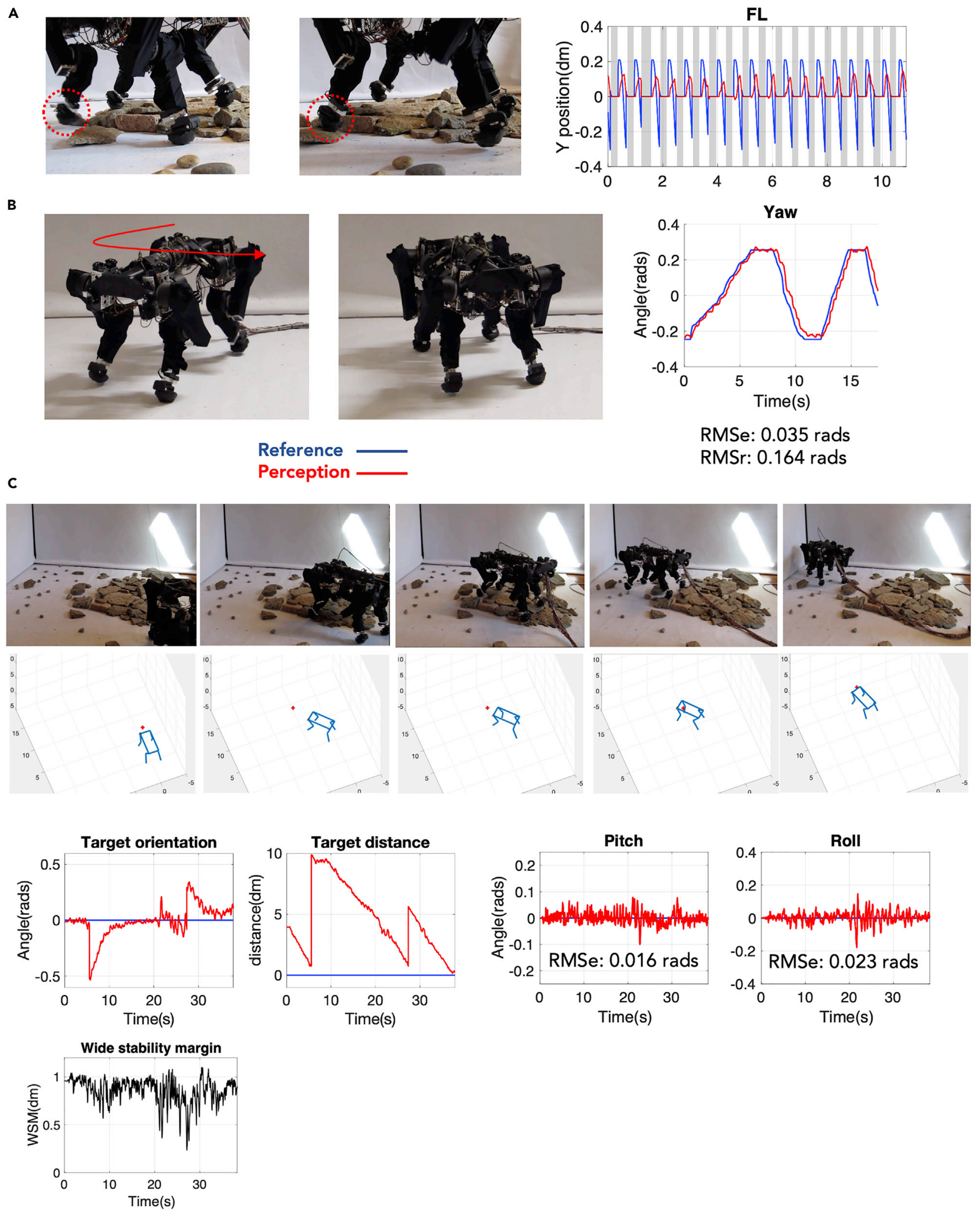

A
FL
Y position(dm)
0.4
0.2
0
-0.2
-0.4
0
2
4
6
8
10
B
Yaw
Angle(rads)
Time(s)
RMSe: 0.035 rads
RMSr: 0.164 rads
Reference
Perception
C
Target orientation
Angle(rads)
Time(s)
Target distance
distance(dm)
Pitch
RMSe: 0.016 rads
Roll
RMSe: 0.023 rads
Wide stability margin
WSM(dm)

Figure 5. Stepping, orientation, and locomotion over rough terrain

(A) During stepping, the feet lift in alternating diagonal pairs. Two images show snapshot of the two major phases of the gait. The left image shows the stage in which the FL and BR legs are up and FR and BL legs are down. The right image shows the opposite stage. Plot shows representative plots of Y (distance from ground) position reference signals (blue) and perceptual signals (red) during stepping for the front left foot during stepping. Gray bars indicate when a given foot controller is off and the leg is being used as part of the trunk control output functions.

(B) During stepping, adjusting the trunk yaw reference signal causes the body orientation to change as the feet lift and reset position.

(C) During stepping, the robot can successfully traverse rough and unstable terrain in pursuit of targets. A target orientation controller at level 4 adjusts the yaw reference signal at level 3, while a target distance controller at level 4 adjusts the front-back trunk position reference signal at level 3. Images are snapshots from a single target pursuit session. The location of the target at the time of each still image is illustrated in the corresponding wire frame diagram below as a red dot. Plots show target orientation, target distance, trunk pitch, trunk roll, and wide stability margin (WSM) measurements for the same session. Because of speed limits on locomotor behavior, orientation and distance errors in these systems reduce relatively slowly compared with lower systems, as illustrated in the target orientation and target distance control plots. The moment of sequence turnover can be seen in the plots as sharp increases in error. Pitch and roll plots illustrate that trunk orientation is continuously defended. WSM is calculated as the distance from the center of gravity to the nearest edge of the support polygon (30). WSM plots illustrate that the center of gravity is successfully maintained above the support polygon.

DISCUSSION

Here we present a legged robot capable of posture control and locomotion. The design is simple, consisting of a hierarchical network of negative feedback control systems in which higher-level systems reach their goals by adjusting the goals of lower systems. The robot can reliably reach a sequence of position targets across rough and unstable terrain. It achieves natural goal-directed locomotion without any training or prior knowledge of the environment, despite significant limitations in motor speed and power, transport delays, and imprecise components.

Unlike other robot control architectures that perform model-based control and planning (Bledt et al., 2018; Bongard et al., 2006; Focchi et al., 2017; Hwangbo et al., 2019; Tsounis et al., 2020), our design generates robust and adaptive goal-directed behavior without requiring a model of the environment, prediction of future states, or learning. The only computations required are operations like subtraction and integration, which can be successfully implemented by analog computing devices.

A notable feature of our design is that it does not require any explicit computation of inverse or forward kinematics and dynamics. Instead of performing such computations based on knowledge of detailed physical laws and predicting how a pre-specified motor command will move effectors, the control modules in our robot simply use negative feedback to control what they sense, as specified by higher-level controllers. This process only requires computing the error signal (discrepancy between perception and reference) and sending it to the appropriate lower systems. The output of each controller simply mirrors and opposes environmental disturbances relative to an internal goal value. The essential computations can be performed by analog circuits or implemented by neural circuits consisting mainly of excitatory and inhibitory neurons.

Finally, since each degree of freedom is handled by a dedicated controller, increasing the degree of freedom only requires an increase in the size of the network. There is no exponential increase in computational demand when the number of joints is significantly increased. Consequently, our design dramatically lowers the computational cost for real-time control with high degrees of freedom.

Comparison with other approaches

Because there is no general consensus on the operating principles of the nervous system, a wide variety of techniques have been used in neurally inspired robotics. One common approach is based on sensorimotor transformations. For example, a robot might exhibit stabilizing postural adjustment in response to body tilt above a threshold value (Ayers and Witting, 2007; Korkmaz et al., 2021) or modify stepping output in response to a particular pattern of sensory input (Kimura et al., 2007; Owaki and Ishiguro, 2017; Schilling et al., 2013). These designs are mostly open loop and rely on internal network dynamics rather than negative feedback to generate behavior. Consequently, they cannot handle unexpected disturbances in any natural environment. The purposive nature of animal behavior is often mistakenly assumed to be a feedforward process in which stimulus input is transformed by the nervous system to produce motor output (Powers, 1978; Yin, 2013, 2020). The concept of a reflex is a classic example of this feedforward assumption in motor control. Yet a reflex only gives the impression of sensorimotor transformation because the actual controlled variables and reference states are not taken into account. Contrary to common assumptions, behavior is not simply "released" or modified by environmental conditions; rather it is actively directed toward accomplishing a set of internally specified and continuously changing goals. In our model, there is no

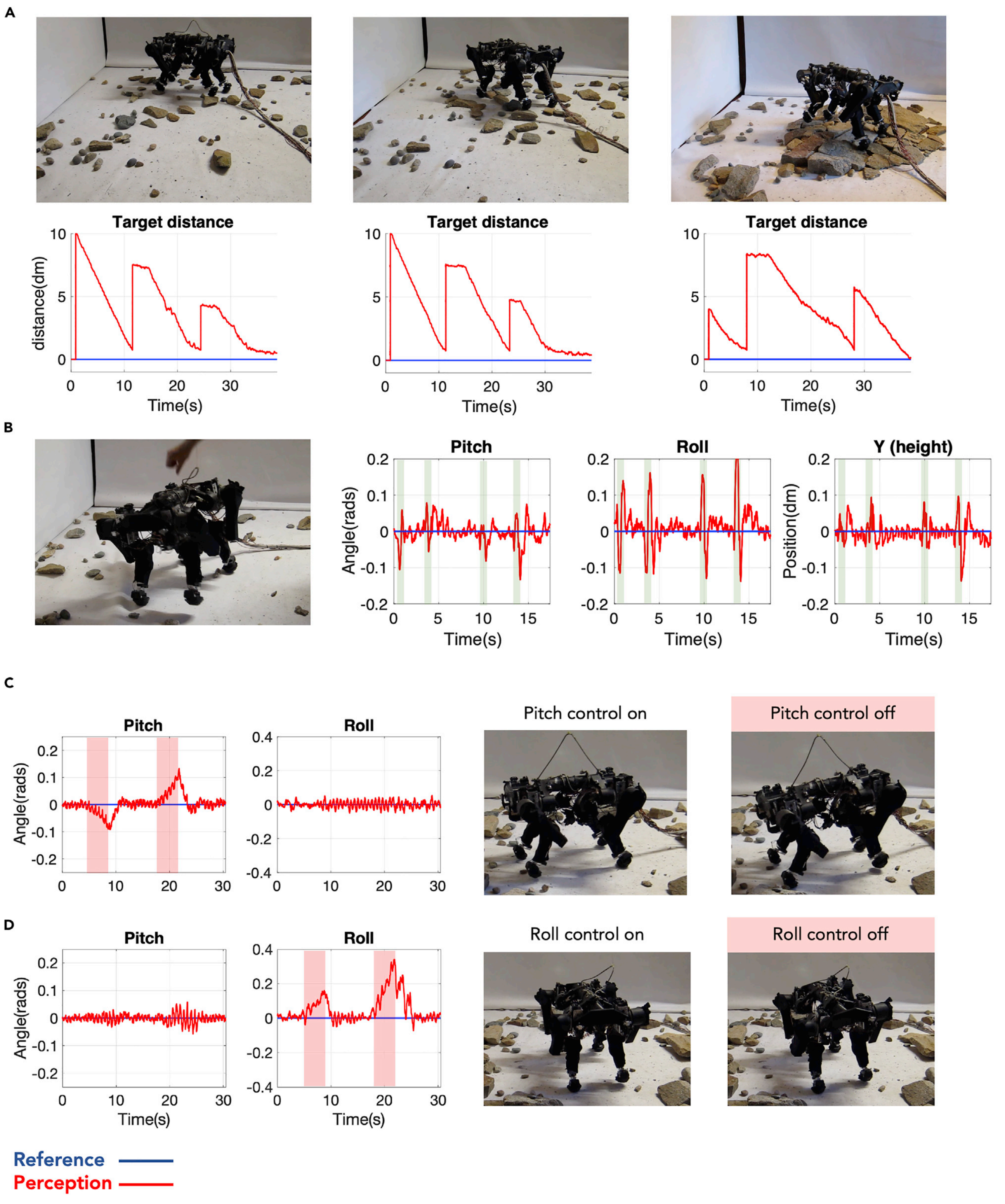


Figure 6. Locomotor control

(A) Three snapshots and corresponding target distance plots for locomotion to three targets across different rough terrains. Blue circles in the snapshots indicate the approximate location of targets. As shown in the plots, the robot hits all three targets by reducing the target distance error towards zero.

Figure 6. *Continued*

(B) A snapshot shows the experimenter pushing the robot on its side, whereas plots show pitch, roll, and trunk height control signals during repeated push disturbances. Red lines are perceptual signals, blue lines are reference signals, and push disturbances are indicated by light green bars. The plots show successful defense of the reference orientations and heights.

(C) Plots showing perceptual and reference signals during a session in which the robot was walking in place when the pitch controller was periodically turned off, as indicated by pink bars. As soon as the controller is turned off, the unpredicted errors caused by stepping itself accumulate and cause the pitch orientation to drift. Turning the controller back on restores the perceptual signal so that it matches the reference signal. Snapshot images to the right show trunk posture when the pitch controller is turned on or off.

(D) Plots and snapshots have the same format as in (C), except that the roll controller is periodically turned off instead of the pitch controller. Roll control can effectively defend against the disturbances generated by stepping.

sensorimotor transformation. To control input relative to a goal by generating variable output, comparison between a sensory input and a reference value is needed rather than a direct transformation of sensory input into motor output.

Unlike traditional neural networks and reinforcement learning (RL), error reduction in our robot is not used for learning but is instead responsible for generating behavior in real time (Barto et al., 1983; LeCun et al., 2015). RL approaches usually require extensive offline training, typically many millions of trials, to establish the correct mapping of sensory input onto motor output (Hassabis et al., 2017). For example, in a simulated legged robot using RL, a readout map is trained using a reward function to transform sensory inputs into modulatory changes to a dynamic oscillator (Arena et al., 2017), or sometimes RL is performed in simulation and the policy is then uploaded to a physical body to generate locomotion (Lee et al., 2020). In contrast, the present approach does not require any training.

Although the presence of a reward function in RL gives the appearance of goal directedness, there is in fact no active pursuit of a behavioral goal, but only shaping of a sensorimotor transformation or state-policy mapping to maximize reward. The reward is never defined with respect to the internal states of the agent. It may be argued that the reward function could provide feedback based on a comparison between actual and desired robot behavior, but such feedback is entirely derived from the perspective of the observer or designer. There is no internal definition of reward and no control of input (Sutton and Barto, 2018). This ''objectivist'' fallacy is responsible for the inappropriate definition of error signals in traditional approaches in RL and optimal control theory, even when control theory is used (Todorov and Jordan, 2002; Yin, 2020). By contrast, in our design the comparison is not performed from the perspective of the observer but from that of the agent, which contains internal reference signals at all levels of the hierarchy. RL architectures, then, are not truly autonomous. Like approaches relying on sensorimotor transformations, they are also open loop, despite their use of an error correction mechanism. Consequently, RL approaches have difficulty dealing with unpredictable disturbances in natural environments. Even after extensive training, the learned policies perform poorly when the environment changes or when disturbances are introduced.

The present approach also differs from model-based control, another approach that has been used in legged robots (Wolpert and Ghahramani, 2000; Wolpert and Kawato, 1998). This approach requires an accurate internal model, which is updated by sensory inputs and used to simulate its dynamics in order to calculate the appropriate movement command. This command can be optimized according to a cost function defining the task goals, such as moving the trunk of the robot to a desired position while also respecting contact configuration and friction constraints (Da et al., 2020). The success of this approach depends on the accuracy of the internal model. In robots with simple and well-characterized bodies, model-based control can be feasible, although it requires sophisticated computations that cannot be achieved by the nervous system (as opposed to scientists trained in symbol manipulation, familiar with physical laws, and using powerful computers capable of performing matrix calculations in real time). Moreover, in biologically realistic legged locomotion, with increased degrees of freedom and unpredictable environments, model-based approaches face insurmountable computational challenges even using the most powerful computers. In contrast, as already mentioned, increasing degrees of freedom does not present significant challenges for our approach.

Advantages of hierarchical control

The benefits of hierarchical architectures have long been known. A recent review acknowledges the advantages of hierarchical motor control but admits that there is currently no plausible model for modular control and communication between levels in the hierarchy (Merel et al., 2019). Here we propose a working model based on the control of input. Our design is directly inspired by the hierarchical organization of the motor

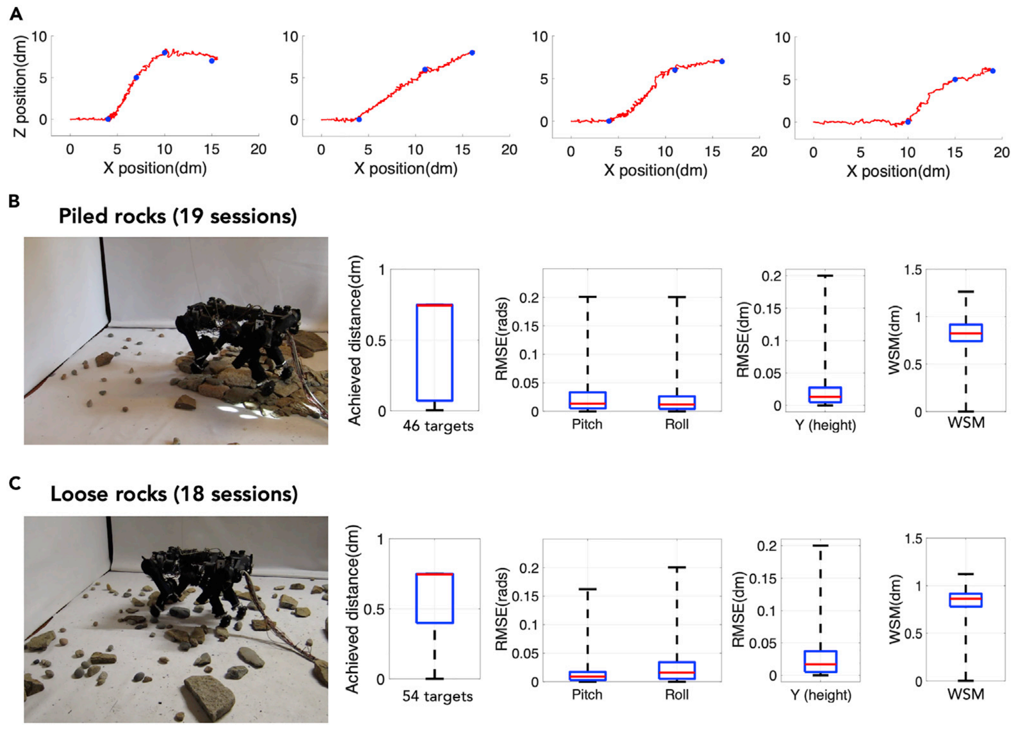


Figure 7. Aggregate performance across rough terrain

(A) Four representative trajectories are shown in red. Blue dots represent target locations.

(B) The leftmost image shows a snapshot example of the piled rock terrain. Boxplots show aggregate performance summaries of 19 sessions across this terrain, including 46 targets in total. From left to right, plots show summary statistics for achieved distance relative to each target, RMS error for trunk pitch, RMS error for trunk roll, RMS error for trunk height, and wide stability margin (WSM). The red line represents the median, the bottom and top edges of the box represent the 25th and 75th percentiles, respectively, and the whiskers extend to the most extreme data points that are not considered outliers.

(C) Eighteen sessions of target pursuit, including 54 targets, across a terrain of loose rocks as shown in the snapshot image. Same format as (B).

system (Grillner et al., 2008; Yin, 2014). In vertebrates, locomotion and posture are accomplished by a hierarchy of neural circuits in which spinal circuits are coordinated by descending reticulospinal projections, which are in turn coordinated by still higher areas including the mesencephalic locomotor region (MLR), tectum, and the cortico-basal ganglia network (Grillner et al., 2008). The efficacy of our design supports input control hierarchies as a working model for how the nervous system generates behavior. Our model is also in accord with extensive experimental work in neuroscience (Barter et al., 2014, 2015a, 2015b). For example, neural activity recorded from the substantia nigra pars reticulata and ventral tegmental area during voluntary movement show strong and continuous correlations with Cartesian head position and 3D head orientation (Barter et al., 2015b; Hughes et al., 2019), respectively, suggesting that these signals could act as reference signals for downstream posture controllers. Reference signals in our robot show a similar close correlation with actual achieved body position (Figures 3, 4, 5A, and 5B), as the robot actively seeks to match them.

Sensory (transport) delays are often cited as a challenge for the use of negative feedback by the nervous system, because they could result in instability (Franklin and Wolpert, 2011; Rack, 2011). However, this view is based on questionable assumptions. As transport delays are present in all control systems, it would be extremely misleading to suggest that they inevitably lead to oscillations. It is more accurate to say that delays could potentially contribute to oscillations, but normally they do not present a problem because the sensory input could be filtered accordingly, the gain adjusted, and the output function sufficiently slowed to prevent oscillations. In our model, integration in the output function is a key feature that promotes stability. Moreover, although biological controllers do not possess or require very high gain individually, the true loop gain in an organism is not due to the properties of a single controller but of a complex hierarchy.

In such a hierarchy, lower systems with shorter sensory delays are able to stably control their own rapidly changing perceptual variables, whereas higher systems with longer sensory delays are built on top of and operate through lower systems. Higher systems are thus usually protected from fast disturbances from the environment, making it much easier to achieve stability.

Summary and conclusions

Unlike conventional approaches in robotics, our control architecture does not require any learned policy or internal model of the body. Every control module, with the exception of the those that adjust the foot position reference signals, independently and continuously seeks to match its perception with its goal value through corrective outputs in real time. Unlike designs that only implement feedback control at the level of joint control and some form of feedforward computation above that to generate behavior (Bledt et al., 2018; Focchi et al., 2017; Kalakrishnan et al., 2010), our design uses negative feedback control at every level of the hierarchy.

In short, the network architecture presented here, inspired by the organization of the nervous system and hierarchical perceptual control, and tested with a working robot, suggests a new and powerful approach in robotics. It provides a foundation not only for future work building testable models of the nervous system but also for the development of intelligent systems more generally.

Limitations of the study

Currently our robot walks slowly, at approximately 8 cm/s, but this does not reflect limitations in the design. The low speed is primarily due to the use of cheap servo motors and 3D printed components, all of which cost less than $ 3000. For example, the motors used can only achieve a foot height of about 2 cm during locomotion. The speed of locomotion can be increased significantly with improved motors and parts. In addition, as our current design has only four levels in the hierarchy, it is not yet a complete model of the whole neural hierarchy. To generate more complex forms of goal-directed behavior, it would be necessary to add higher levels and additional sensors.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability
- EXPERIMENTAL MODEL AND SUBJECT DETAILS
- METHOD DETAILS
  - Testing controllers
- QUANTIFICATION AND STATISTICAL ANALYSIS

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.isci.2021.102948.

ACKNOWLEDGMENTS

We would like to thank Konstantin Bakhurin and Ryan Hughes for helpful discussion during the writing of this paper and comments on the manuscript.

AUTHOR CONTRIBUTIONS

H.H.Y. developed the theory of neural hierarchical control and supervised the research project. J.W.B. designed, built, and tested the robot. J.W.B. and H.H.Y. wrote the manuscript.

DECLARATION OF INTERESTS

An invention and patent application by J.W.B. with partial overlap was filed. H.H.Y. declares no competing interests.

Received: February 18, 2021
Revised: May 25, 2021
Accepted: August 2, 2021
Published: September 24, 2021

REFERENCES

Arena, E., Arena, P., Strauss, R., and Patané, L. (2017). Motor-skill learning in an insect inspired neuro-computational control system. Front. Neurorobotics *11*, 12.

Ayers, J., and Witting, J. (2007). Biomimetic approaches to the control of underwater walking machines. Philosophical Trans. R. Soc. A: Math. Phys. Eng. Sci. *365*, 273–295.

Barter, J.W., Castro, S., Sukharnikova, T., Rossi, M.A., and Yin, H.H. (2014). The role of the substantia nigra in posture control. Eur. J. Neurosci. *39* (9), 1465–1473.

Barter, J., Li, S., Lu, D., Rossi, M., Bartholomew, R., Shoemaker, C.T., Salas-Meza, D., Gaidis, E., and Yin, H.H. (2015a). Beyond reward prediction errors: the role of dopamine in movement kinematics. Front. Integr. Neurosci. *9*, 39.

Barter, J.W., Li, S., Sukharnikova, T., Rossi, M.A., Bartholomew, R.A., and Yin, H.H. (2015b). Basal ganglia outputs map instantaneous position coordinates during behavior. J. Neurosci. *35*, 2703–2716.

Barto, A.G., Sutton, R.S., and Anderson, C.W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Trans. Syst. Man Cybern. *13*, 834–846.

Bernstein, N. (1967). The Coordination and Regulation of Movements (Pergamon Press).

Bledt, G., Powell, M.J., Katz, B., Di Carlo, J., Wensing, P.M., and Kim, S. (2018). Mit cheetah 3: design and control of a robust, dynamic quadruped robot. Paper presented at: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE).

Bongard, J., Zykov, V., and Lipson, H. (2006). Resilient machines through continuous self-modeling. Science *314*, 1118–1121.

Da, X., Xie, Z., Hoeller, D., Boots, B., Anandkumar, A., Zhu, Y., Babich, B., and Garg, A. (2020). Learning a Contact-Adaptive Controller for Robust, Efficient Legged Locomotion, arXiv:200910019.

Focchi, M., Del Prete, A., Havoutis, I., Featherstone, R., Caldwell, D.G., and Semini, C. (2017). High-slope terrain locomotion for torque-controlled quadruped robots. Autonomous Robots *41*, 259–272.

Franklin, D.W., and Wolpert, D.M. (2011). Computational mechanisms of sensorimotor control. Neuron *72*, 425–442.

Grillner, S. (2011). Control of locomotion in bipeds, tetrapods, and fish. Compr. Physiol. 1179–1236.

Grillner, S., Wallen, P., Saitoh, K., Kozlov, A., and Robertson, B. (2008). Neural bases of goal-directed locomotion in vertebrates–an overview. Brain Res. Rev. *57*, 2–12.

Hassabis, D., Kumaran, D., Summerfield, C., and Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. Neuron *95*, 245–258.

Hughes, R.N., Watson, G.D., Petter, E.A., Kim, N., Bakhurin, K.I., and Yin, H.H. (2019). Precise coordination of three-dimensional rotational kinematics by ventral tegmental area GABAergic neurons. Curr. Biol. *29*, 3244–3255.e3244.

Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., and Schaal, S. (2010). Fast, robust quadruped locomotion over challenging terrain. Paper presented at: 2010 IEEE International Conference on Robotics and Automation (IEEE).

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., and Hutter, M. (2019). Learning agile and dynamic motor skills for legged robots. Sci. Robotics *4*, eaau5872.

Kimura, H., Fukuoka, Y., and Cohen, A.H. (2007). Biologically inspired adaptive walking of a quadruped robot. Philosophical Trans. R. Soc. A: Math. Phys. Eng. Sci. *365*, 153–170.

Korkmaz, D., Ozmen Koca, G., Li, G., Bal, C., Ay, M., and Akpolat, Z.H. (2021). Locomotion control of a biomimetic robotic fish based on closed loop sensory feedback CPG model. J. Mar. Eng. Technology *20*, 125–137.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature *521*, 436–444.

Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., and Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. Sci. Robotics *5*, eabc5986.

McCrea, D.A., and Rybak, I.A. (2007). Modeling the mammalian locomotor CPG: insights from mistakes and perturbations. Prog. Brain Res. *165*, 235–253.

McCrea, D.A., and Rybak, I.A. (2008). Organization of mammalian locomotor rhythm and pattern generation. Brain Res. Rev. *57*, 134–146.

Merel, J., Botvinick, M., and Wayne, G. (2019). Hierarchical motor control in mammals and machines. Nat. Commun. *10*, 1–12.

Owaki, D., and Ishiguro, A. (2017). A quadruped robot exhibiting spontaneous gait transitions from walking to trotting to galloping. Scientific Rep. *7*, 1–10.

Powers, W.T. (1973). Behavior: Control of Perception (Benchmark Publications).

Powers, W.T. (1978). Quantitative analysis of purposive systems. Psychol. Rev. *85*, 417–435.

Powers, W.T., Clark, R.K., and McFarland, R.L. (1960). A general feedback theory of human behavior. Perceptual Mot. Skills *11*, 71–88.

Rack, P.M. (2011). Limitations of somatosensory feedback in control of posture and movement. Compr. Physiol. 229–256.

Schilling, M., Paskarbeit, J., Hoinville, T., Hüffmeier, A., Schneider, A., Schmitz, J., and Cruse, H. (2013). A hexapod walker using a heterarchical architecture for action selection. Front. Comput. Neurosci. *7*, 126.

Sutton, R.S., and Barto, A.G. (2018). Reinforcement Learning: An Introduction (MIT press).

Todorov, E., and Jordan, M.I. (2002). Optimal feedback control as a theory of motor coordination. Nat. Neurosci. *5*, 1226–1235.

Tsounis, V., Alge, M., Lee, J., Farshidian, F., and Hutter, M. (2020). Deepgait: planning and control of quadrupedal gaits using deep reinforcement learning. IEEE Robotics Automation Lett. *5*, 3699–3706.

Wolpert, D.M., and Ghahramani, Z. (2000). Computational principles of movement neuroscience. Nat. Neurosci. *3*, 1212–1217.

Wolpert, D.M., and Kawato, M. (1998). Multiple paired forward and inverse models for motor control. Neural Networks *11*, 1317–1329.

Yin, H.H. (2013). Restoring purpose in behavior. In Computational and Robotic Models of the Hierarchical Organization of Behavior (Springer), pp. 319–347.

Yin, H.H. (2014). How basal ganglia outputs generate behavior. Adv. Neurosci. *2014*, 768313.

Yin, H.H. (2017). The basal ganglia in action. Neuroscientist *23*, 299–313.

Yin, H.H. (2020). The crisis in neuroscience. In The Interdisciplinary Handbook of Perceptual Control Theory (Elsevier), pp. 23–48.

STAR★METHODS

KEY RESOURCES TABLE

RAEGENT or RESOURCE	SOURCE	IDENTIFIER
Other		
Maestro 24 servo control board	Pololu	Cat# 1356
NI USB-6225 analog to digital converter	National Instruments	Cat# 197294A-01L
7954SH servo motors	Hitec	Cat# 37954A
Servo gear drives	Servo City	Cat# CM-400-180-3-24
		Cat# CM-400-180-2-24

RESOURCE AVAILABILITY

Lead contact

- Further information and requests should be directed to and will be fulfilled by the lead contact, Henry Yin (hy43@duke.edu).

Materials availability

- This study did not generate new unique reagents.

Data and code availability

- All data from this study may be available from the corresponding author upon reasonable request.
- Data analysis was performed using standard MATLAB functions. This paper does not report original code.

EXPERIMENTAL MODEL AND SUBJECT DETAILS

The sole subject from which data was collected is the robot described here. The robot is a legged quadruped with 3 motor actuated joints per leg. The body is made from 3D printed parts and light aluminum channeling. Motors are standard size hobby servos (HS-7954SH, Hitec) with gear drives (Servo City). Knees used a 2:1 gear reduction while shoulder motors used a 3.8:1 gear reduction. These joints are not easily back-drivable. The body weighs 7.8 kg and measures approximately 6 dm long by 4 dm wide. Upper leg joints are located 4.5 dm apart lengthwise and 1.9 dm apart widthwise.

METHOD DETAILS

To help support the body weight and reduce gear play, a torsion spring is installed in each joint to oppose the effect of gravity. Torsion springs transmit force by gears into the drive train of each joint. Torsion gears add force in parallel with servomotors.

Each joint is equipped with an analog position sensor and an analog torque sensor between the motor and the movable body part. Joint angular velocity is sensed by taking the first derivative of joint position from one step of the control loop to the next. Each foot is equipped with an analog force sensor for detecting ground reaction forces. For a vestibular system, the trunk of the robot houses an analog Inertial Measurement Unit (IMU) consisting of a 2-axis accelerometer and two gyros for sensing orientation of the body in space. Sensory inputs are communicated from the body to the computer by a shielded multi conductor cable and read into a standard desktop PC as analog voltages through a National Instruments Box (NI USB-6225). Motor outputs are commanded from the PC through a USB servo control board (Maestro 24, Pololu). There is a latency of approximately 75 ms between a command being issued and movement by the motor. The robot uses a tethered power supply (7.4 V and 20a).

Torque is controlled by converting the torque controller output into a motor position through a virtual driven damped torsion spring, a feature which was included to smooth outputs and improve compliance to small high frequency disturbances. Although virtual spring compliance makes the behavior of effectors more dynamic and unpredictable, this does not pose a problem for our approach which reduces perceptual errors regardless of their source. Damping constant and spring constant values were selected to generate a stiff spring system exhibiting small compressions relative to the overall movement of the joint. The spring has a resting length of zero. The virtual spring is represented by the following equation:

$$f = -k(x - X) - c\frac{dx}{dt} + S_f$$

Here, X is the position of the virtual driver determined by the output of the torque controller. x is the position of the virtual mass connected by a spring to the driver, which determines the servomotor reference value. By adjusting X, we adjust the length of the virtual spring and thus the force acting on x. k is the spring constant, c is the damping constant, and S_f is the force sensed by the torque sensor. The total force, f, acting on the effector at a given time determines the acceleration of x.

Cartesian coordinates are calculated from joint angles and segment lengths using trigonometry. For each leg, we calculate the foot and knee position in Cartesian space with the shoulder as the origin. To position these body parts in egocentric space, we then translate these values by adding or subtracting fixed offsets along the horizontal plane so that each shoulder is positioned the correct measured distance from the origin at the center of the trunk. To achieve position relative to gravity, we then rotate all body points around the origin according to the pitch and roll angles sensed by the IMU.

The controllers operate according to the following equations:

$$e = r - p$$
$$u = K_p e + K_d\frac{de}{dt}$$

Here, e is error, r is reference signal, and p is perceptual signal, o is output signal, K_p is proportional gain, and K_d is derivative gain.

The gain scheduler interfacing between level 3 (posture) and level 2 (joint angular velocity) systems functions to dynamically scale the outputs from level 3 systems to produce posture velocities that retain their linear relationship with those outputs regardless of the current body posture. This feature is not essential; as long as the PD gains are appropriate, and as long as we convert a given posture error into a set of joint velocity reference signals which oppose that error, then the error will approach zero. However, gain scheduling is useful for stabilizing large and fast outputs during locomotion. The gain scheduler works as follows: from the current perceived body posture, PD outputs from active level 3 controllers are converted into movements in the opposite direction as the errors that caused them, to generate a new 'virtual' posture. This new posture is then subtracted from the current perceived posture to calculate the joint angle changes between them. These angles determine proportionate velocity reference signals sent to corresponding level 2 joint angular velocity controllers.

As a stabilizing mechanism to absorb disturbances and prevent overreaction, we include an error threshold for trunk position and orientation controllers at level 3. If this error threshold is exceeded, such as in response to a push, then the corresponding reference signal moves so as to keep the error below the threshold value. The reference signals continuously and slowly move back to their original positions.

PD gains were tuned by hand to balance stability and speed while meeting the overall performance standards outlined below. When tuning a layer of the hierarchy, the higher layers were left inactive and the reference signals were adjusted by the experimenter to assess performance. Once tuned, no further adjustments are needed for the robot to adapt to new environments. The key performance criterion for the network as a whole is the ability to reliably reach locomotor targets in spite of unpredictable disturbances. Secondary performance criteria include stability and speed in lower controllers during locomotion, such as trunk and foot positioning and joint control. Joint controllers of a particular type shared the same PD gains. Each individual controller above level 2 had a unique set of gains.

Testing controllers

To test foot position controllers, the robot was suspended hanging and tipped by the experimenter while foot controllers for a single leg were active. To test trunk orientation controllers, the robot was placed, with all trunk position and orientation controllers active, on a platform which was randomly tipped by the experimenter along pitch and roll planes. To test the viability of the proposed network architecture for locomotion, the robot was required to navigate across terrains of unstable and loose and piled rocks to achieve a sequence of targets (Figures 6A and 7A). 48 total sessions were performed across loose rock terrain shown in Figure 7B, of which only 7 (15%) ended in a fall. Of these 7 falls, 2 were caused by a foot becoming stuck and 5 were caused by trunk movements that were too fast for the feet to catch, typically when at least one foot was planted on an unstable rock. Internal signals were recorded from 19 of these sessions with no falls. These data show that the robot achieved close target distance (distance between robot and target, with 0 being perfect overlap) for all targets, and good trunk control and wide stability margin (Figure 7B). 25 sessions were performed across the piled rock terrain shown in Figure 7C, of which 3 (12%) ended in a fall. Of these 3 falls, 2 were caused by a foot becoming stuck and 1 was caused by trunk movements that were too fast for the feet to catch. Internal signals were recorded from 18 of these sessions with no falls. Analysis of these data show good achieved distance relative to targets and good trunk control and stability (Figure 7C). Piled rock obstacles in these sessions formed sloped terrains with slopes of between 5 and 15°, and step features up to about 0.2dm. Performance dropped off quickly as the step feature height and slope were increased beyond this level. This performance drop-off is largely due to its underpowered motors, which achieve insufficient step height. In later versions, we hope to address this limitation so that it is possible to perform meaningful performance comparisons with other robot control architectures.

QUANTIFICATION AND STATISTICAL ANALYSIS

Summary statistics bar plots in Figure 7 were generated in MATLAB using the *boxplot* function.