# Hierarchical generative modelling for autonomous robots

**Kai Yuan** ( kai.yuan@ed.ac.uk )
The University of Edinburgh
**Noor Sajid**
University College London
**Karl Friston**
University College London https://orcid.org/0000-0001-7984-8909
**Zhibin Li**
The University of Edinburgh

**Article**

**Keywords:** autonomous robotic operations, hierarchical generative modelling, autonomous task completion

**Posted Date:** November 15th, 2021

**DOI:** https://doi.org/10.21203/rs.3.rs-965852/v1

# Hierarchical generative modelling for autonomous robots

Kai Yuan[1,3], Noor Sajid[2,3], Karl Friston[2] and Zhibin Li[1*]

[1] Edinburgh Centre for Robotics, The University of Edinburgh, UK
[2] Wellcome Centre for Human Neuroimaging, Queen Square Institute of Neurology, University College London, UK
[3] These authors contributed equally

* Corresponding author: zhibin.li@ed.ac.uk

**ABSTRACT**

Humans can produce complex movements when interacting with their surroundings. This relies on the planning of various movements and subsequent execution. In this paper, we investigated this fundamental aspect of motor control in the setting of autonomous robotic operations. We consider hierarchical generative modelling—for autonomous task completion—that mimics the deep temporal architecture of human motor control. Here, temporal depth refers to the nested time scales at which successive levels of a forward or generative model unfold: for example, the apprehension and delivery of an object requires both a global plan that contextualises the fast coordination of multiple local limb movements. This separation of temporal scales can also be motivated from a robotics and control perspective. Specifically, to ensure versatile sensorimotor control, it is necessary to hierarchically structure high-level planning and low-level motor control of individual limbs. We use numerical experiments to establish the efficacy of this formulation and demonstrate how a humanoid robot can autonomously solve a complex task requiring locomotion, manipulation, and grasping, using a hierarchical generative model. In particular, the humanoid robot can retrieve and deliver a box, open and walk through a door to reach the final destination. Our approach, and experiments, illustrate the effectiveness of using human-inspired motor control algorithms, which provide a scalable hierarchical architecture for autonomous performance of complex goal-directed tasks.

# 1. INTRODUCTION

Humans can control their bodies to produce intricate motor behaviours that are aligned with their objectives, e.g., opening the door or moving a box. These tasks rely on the coordination of two distinct processes: motor planning and execution [1-4]. To realise this coordination, human motor control unfolds at nested time scales at different levels of the neuronal hierarchy [5,6], e.g., a high-level plan to arrive at a particular place can entail multiple, individual, reflexive low-level limb movements for walking. Similar distinctions have been introduced in the robotics and control literature [7], with promising results of combining robot control with learnt motor movements. However, control and planning are often designed manually and separately, with little or no feedback between the two [8]. This structural separation limits the performance of the robot and places greater demands on human involvement at the deployment stage, even under controlled conditions.

Given these principles, we postulate that robotics systems should consider motor control as a consequence or outcome of a (learnt) hierarchical generative model, which can optimise and adapt future actions in uncertain environments. This proposal inherits from hierarchical formulations of motor control and ensuing planning or control as (active) inference [9-21]. Briefly, hierarchical generative models are a description of how sensory observations are generated, i.e., encodings of sensorimotor relationships relevant for motor control [22,23]. This implies that autonomous systems can be conceptualised in terms of hierarchical motor control, capable of dealing with complex and diverse tasks [24].

Practically, hierarchical generative models speak to a spatiotemporal separation of planning and motor control that assures functional integration [23,25,26]. The requisite architectures can be regarded as a series of distinct levels that provide appropriate motor control [27](Figure 1). The lowest level predicts the proprioceptive signals—generated using a forward model of the mechanics—and the kinetics that undergirds motor execution. This kinetics can be regarded as realising and equilibrium position or desired set point, without explicit modelling of task dynamics (c.f., the equilibrium point hypothesis) [28,29]. The level above generates the necessary sequence of fixed points, that are realised by the lower

level. This sequence speaks to the stability control that a human has over limbs, to perambulate in an upright manner over, say, a centre of gravity. The highest level then pertains to planning [25,30]. Here, different states represent endpoints of an agent's plan, e.g., move a box from a table to another.

To verify this proposition, we introduce a hierarchal generative model for autonomous robotic operations using a learning-based scheme. Our model has three distinct levels for planning and motor generation. The first rests on the equilibrium point hypothesis[1]; namely, a forward model that predicts the proprioceptive inputs reporting a desired movement, where reflexes realise the descending predictions or set points. Practically, the ensuing hierarchical model was optimised simultaneously at all levels. However, only the middle level planner had the access to state feedback, which allowed for a particular type of factorisation (i.e., functional specialisation) in our generative model. We leave further details for later sections.

Using this generative model, we focused on solving tasks that comprised sequential, conditional decision-making in two scenarios. In the first scenario, the robot needed to reach a goal location by opening a closed door. The door only opened when a particular button was pressed. Here, the planner needed to coordinate both its legs and arms to (i) walk towards the door, (ii) and upon reaching it to move its arms to press the button, and (iii) finally walk through the door to reach the goal location. Conversely, the second scenario required the robot to carry a box from one table to another. Here, the planner needed to prescribe the sequence of approaching and picking up the box, walking towards the second table, and placing the box upon arrival. For both scenarios, our algorithm could perform coherent locomotion, manipulation, and grasping movements similar to humans, and therefore successfully solve these complex tasks.

The paper is organised as follows. In Section 2, we describe the realisations of two hierarchical generative models for motor control: human motor control and our robotic system capable of autonomous operations. The robotic system instantiates an implicit hierarchical generative model for motor control, which entails a bidirectional propagation of information between different levels of the

[1] Briefly, the equilibrium point hypothesis states that all movements are generated by the nervous system through a gradual transition of equilibrium points along a desired trajectory[28,29].

generative model. Next, we show that the ensuing scheme can perform tasks remarkably similar to humans (Section 3). In Section 4, we discuss the effectiveness of our hierarchical generative model, how it may benefit potential applications, such as humanoids in disaster response scenarios, and provide an outlook for future work. Lastly, in Section 5, we provide details of our implementation.

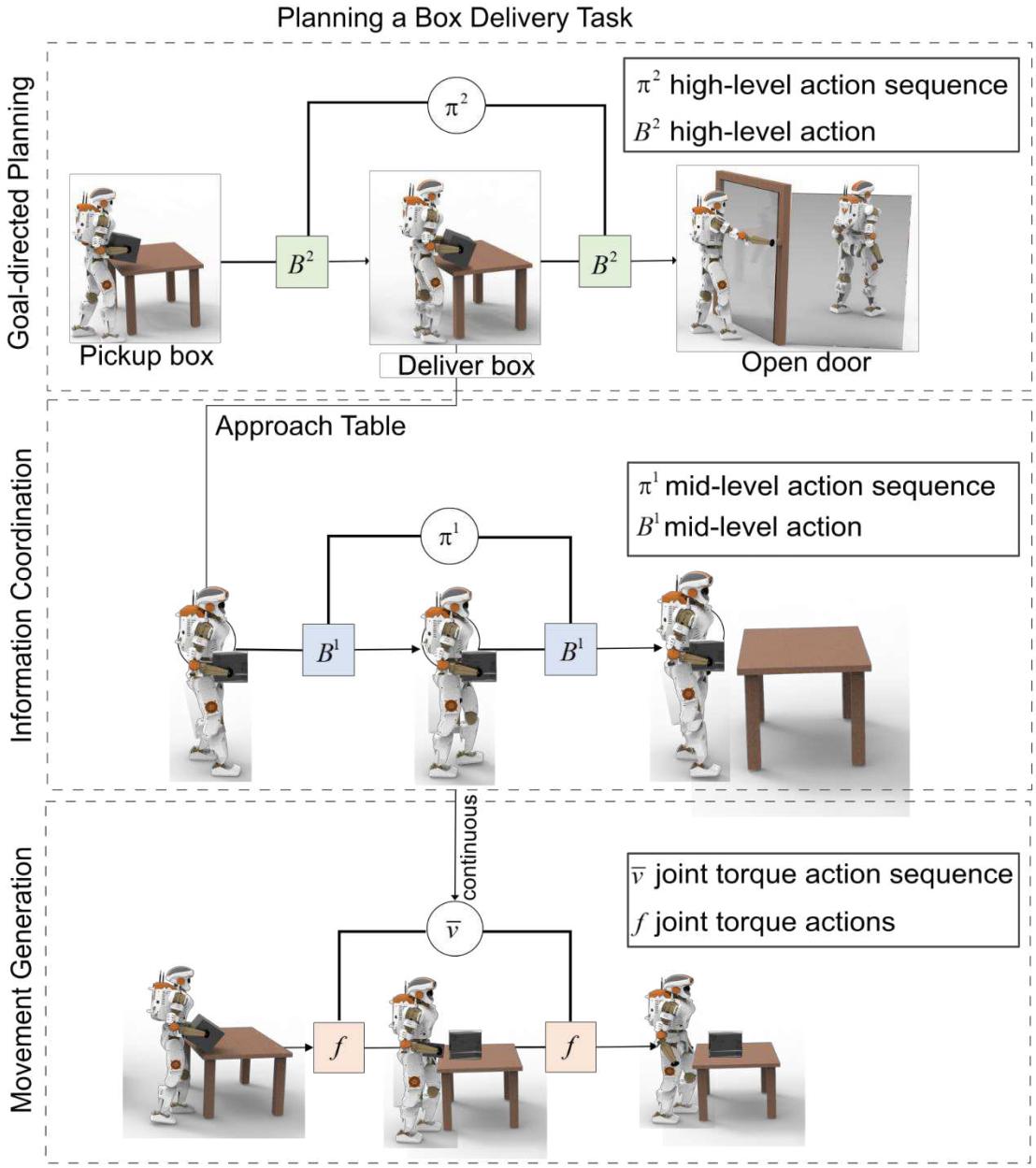


Figure 1: Pictorial representation of a hierarchical generative model for moving boxes – a form of motor control. A generative model represents the conditional dependencies between states and how they cause outcomes. For simplicity, we express this as filled squares that denote actions, and circles that represent action sequences. The key aspect of this model is its hierarchical structure that represents sequences of action over time. Here, actions at higher levels generate the initial actions for lower levels–that then unfold to generate a sequence of actions: c.f., associative chaining. Crucially, lower levels cycle over a

sequence for each transition of the level above. It is this scheduling that endows the model with a deep temporal structure. Particularly, planning (first row; highest level) to "deliver the box" generates the actions for the information coordination level (second row; middle level) i.e., "movement towards the table". This in turn determines the initial actions for movement generation (third row; lowest level) of arms to 'place the box' on the table.

# 2. HIERARCHICAL GENERATIVE MODELS FOR MOTOR CONTROL

We hypothesise that motor control is a natural outcome of hierarchical generative modelling (Figure 1), in particular, generative models that include the consequences of action. Hierarchical generative models are descriptions of how sensory observations are generated – expressed as a joint distribution over the unobservable causes of observable sensory input. These models can be factorised to represent particular conditional dependencies that formalise sensorimotor contingencies [22,23]. Here, causes of proprioceptive input can be predicted by fitting the model to observed sensory data [31]. This provides an explainable and interpretable specification of decision-making within the robot, namely, planning and control as inference. Interestingly, having a generative model gives for free the five core principles of hierarchical motor control introduced in Merel et al., 2019 (see Table 1).

**Table 1.** Summary of the key principles of hierarchical motor control[24], with exemplar realisations in human motor control and our robotic system. We omit the principle of modular objectives here (sub-systems trained to optimise specific objectives distinct from the global task objective) because a factorised generative model architecture leads to distinct factor specific objectives at each level in the hierarchy.

| Principle | Description | Hierarchical generative models | Human motor control | Our robotics system for autonomous operations |
|---|---|---|---|---|
| Information factorisation | Different information is processed by distinct sub-systems. | Factorised distribution of appropriate latent states within the generative model. | Different sensory signals are routed to different parts in the hierarchy, e.g., what and where streams. These neuronal pathways can be characterised as factorised states responsible for sub-systems. | Only task-relevant sensory signals are used by individual levels, with irrelevant states hidden across levels. This speaks to an explicit factorisation of sensory signals and which parts of the system have access to them. |
| Partial autonomy | Lower hierarchical levels can semi-autonomously | The result of factorising state space into multiple levels can independently accomplish sub-goals at a | Semi-autonomous coordination of joint movement at lower levels (i.e., brainstem | Full autonomy and stability guaranteed at individual levels. Explicitly, we introduce stable mid-level |

| | | | | |
|---|---|---|---|---|
| | produce outputs with minimum input from levels above. | (relatively) fast temporal scale. | and spinal cord). These operate at a faster temporal scale and do not require continuous input for higher levels. | and low-level motions for random higher-level inputs. This ensures that lower levels can independently perform fast movements. |
| Amortised control | Re-execute appropriate behaviours rapidly using learnt movements. | Learnt probability distributions that parameterise this generative model can be used for amortised control. That allows for habitual control based on previously learnt distributions. | The cerebellum is responsible for amortised control of deliberative and goal-directed behaviours, evoking fast habitual control for repeated actions. | The system learnt policies (i.e., action-state mappings) that provide habitual control for rapidly re-executing appropriate actions. |
| Multi-joint coordination | Degenerate coupling of different components operating as a whole for motor control. | Result of state factorisations that introduce flexible mapping across and within each level. | Different neuronal ensembles have distinct influences e.g., the red nucleus controls movements of the arms. Much like factorised states, these neuronal ensembles come together to produce intricate movements. | The system is equipped with multiple sub-structures (or policy mappings) that are responsible for specific actuator movement. Together these come across, and within levels, to produce particular motor movements. |
| Temporal abstraction | Abstraction of time across hierarchical levels. | A feature of hierarchical generative models, where higher levels evolve slower than and constrain, the level below. | Different levels evolve at different temporal and spatial scales, with the primary motor cortex responsible for planning (slow timescale) and spinal cord responsible for generation (fast timescale) | The three levels of the system evolve at different temporal scales – much like any hierarchical generative model. The high-level planning is at a slow timescale, mid-level stability control at medium timescale, and low-level joint control at a fast timescale. |

The above considerations speak to the relevance of hierarchical generative models for motor control. In the following, we describe the realisations in two domains: human and robotics (Figure 2).

## GENERATIVE MODELS OF HUMAN MOTOR CONTROL

The inversion of forward models—that underwrites human motor control—generates continuous proprioceptive predictions at the lowest level and propagates information to the highest levels that are responsible for planning. Higher levels then pass messages down the hierarchy to generate particular movements, via reflexive realisation of proprioceptive predictions (Figure 2). At the lowest

(sensorimotor) level, these proprioceptive predictions are usually multisensory. For simplicity, we limit ourselves to motor control by considering predictions of primary afferent signals from muscles.

The generative model's lowest (and fast-evolving) level includes the spinal cord and the brainstem. The local (factorised) circuits in these regions are responsible for organising movements, given descending predictions from different higher levels. Exemplar projections, from upper motor neurons in the cortex, control movement indirectly via pathways that project to the brainstem motor control centres, which, in turn, project to the local organising circuits in the brainstem and the spinal cord. Accordingly, these areas are responsible for evaluating the discrepancy between the proprioceptive inputs (primary afferents) and descending predictions of these signals, to drive muscle contraction via classical motor reflexes and accompanying musculoskeletal mechanics [25,32]. It is generally thought that the above systems coordinate multiple joint movements semi-autonomously over time [33,342]. At this level, different neuronal ensembles have distinct, partially autonomous influences, e.g., the red nucleus controls movements of the arms.

At intermediate levels of this generative model, one could consider the role of the cerebellum and the basal ganglia. The cerebellum receives ascending inputs from the spinal cord, and other areas, and integrates these to fine-tune motor activity. In other words, it does not initiate movement, but contributes to its coordination, precision, and speed, through fast non-deliberative mode of operation. Therefore, it can be thought of as being responsible for amortised (habitual) control of motor behaviour, which is characterised by subcortical and cortical interactions [35-39]. The cerebellum receives information from the motor cortex, processes this information, and sends motor impulses to skeletal muscles (via the spinal cord). Conversely, the basal ganglia is responsible for motor program selection, informed by the thalamus and the motor cortex [40]—appropriately orchestrating lower levels of the generative model (spinal cord via the brainstem). The basal ganglia are generally thought to support learning and action selection [41].

[2] These trajectories are a succession of fixed points such that each fixed point is realised in a specified (e.g., theta) cycle over a particular time period. Then motor sequence can be regarded as filling in the gaps between a series of fixed points, where each fixed points is specified by the level above in very scheduled saltatory discrete fashion.

Higher levels of the generative model include the cerebral cortex and the thalamus. The thalamus receives projections from the cerebellum (related to movement and sensory stimulation) and the basal ganglia (related to wilful movements), and the thalamus projects directly to the primary motor and premotor association cortices [42]. The cortex has access to factorised sensory streams of exteroceptive, interoceptive and proprioceptive signals (e.g., visual, auditory, somatosensory, etc) and hence can coordinate, contextualise or override habitual control elaborated in lower levels. Specifically, the primary motor cortex is responsible for deliberative planning, control, and execution of voluntary movements, for example, when learning a new motor skill prior to its habitisation or amortisation.

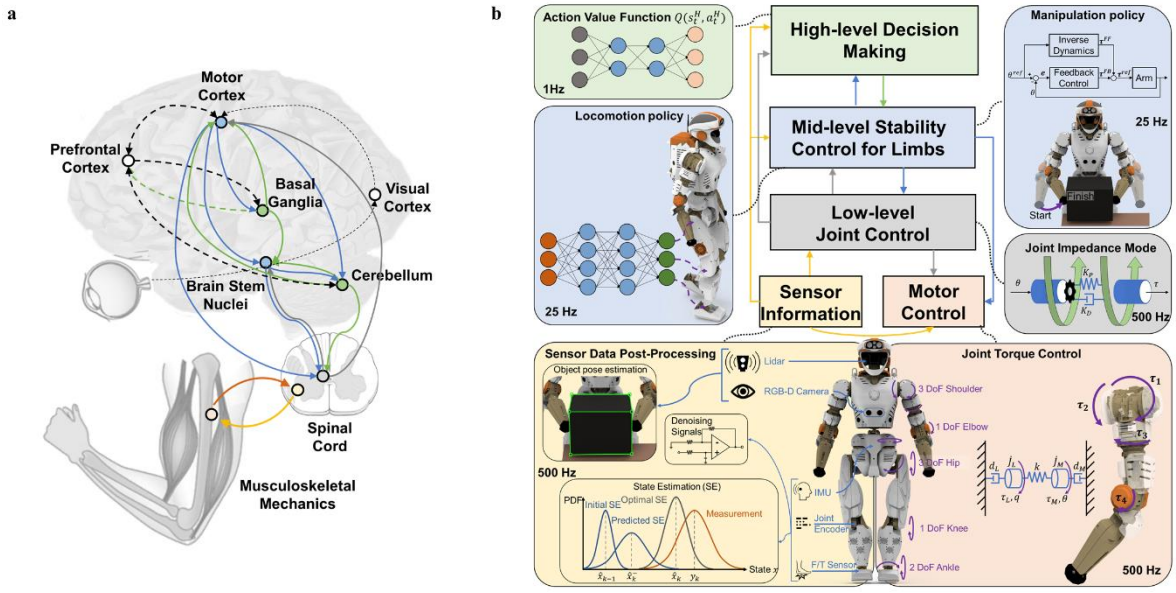


**Figure 2:** Algorithmic realisations of hierarchical control as inference. Panel A presents a schematic of the generative model that underwrites human motor control, and Panel B depicts the implicit generative model for a robotics system. The green nodes in Panel A and green boxes in Panel B refer to the highest levels of human motor control and our implicit generative model respectively. In the generative model, high-level decision making is realised as a neural network learned through Deep Reinforcement Learning. The blue nodes in Panel A correspond to the middle level of human motor control and the blue boxes in Panel B are intermediate level realisations, implemented as Deep Neural Network policy learned through Deep Reinforcement Learning for Locomotion and Inverse Kinematics and Dynamics policy for Manipulation. On the lowest level, depicted in grey nodes and boxes, a joint impedance controller calculates the torques required for the actuation of the robot. Yellow and light red denote sensor information and motor control respectively.

## IMPLICIT HIERARCHICAL GENERATIVE MODEL FOR A ROBOTICS SYSTEM

Following the key principles of hierarchical motor control (Table 1) and the generative model in Figure 1, we have constructed a generative model for a humanoid robot comprising three levels: high-level decision making, mid-level stability control, and low-level joint control (Figure 2). This hierarchical architecture rests on conditional independencies that result in factorised message passing between hierarchical levels[3]. For example, the mid-level locomotion policy only has access to the lower joint states and goals. Here, the temporal depth and structure motor planning is an outcome of specifying a hierarchical generative model, where level-specific policies are evaluated at different timescales. This involves each level assimilating (fast) evidence from the level below, in a way that is contextualised or selected by (slow) constraints, afforded by the level above.

In this example, the highest planning level, evolving at the slowest rate, selects an appropriate sequence of limb movements, which are needed to complete a particular sub-task. It decides where the hands should be and what direction to go. Practically, deep reinforcement learning (RL) is used to learn the correct generation of Cartesian position commands for the mid-level stability control.

These planning targets are realised at the level below that regulates the balance and stability of the robot during manipulation and locomotion. Manipulation is instantiated as a minimum-jerk model-predictive controller that moves the arms to the target positions provided by the high-level policy. Locomotion is implemented as a learnt policy, via deep RL, that coordinates legs—with twelve degrees of freedom—to reach the destination predicted by the higher level. Both of these kinds of policies are designed to ensure that infeasible set points—from the high level—are corrected for the mid-level stability control so that only stable joint target commands are supplied to the low-level joint controller.

Despite receiving inputs from other levels, each level has partial autonomy over its final predictions and goal. Furthermore, multi-joint coordination is realised by learning a policy that coordinates all joints

[3] It should be noted that the implementation used in our simulations is not formulated explicitly in terms of message passing or belief updating (which would usually be articulated in terms of Bayesian filtering for continuous states and belief propagation or variational message passing for discrete states). However, there exists an interpretation of the scheme in terms of expected states of the environment causing sensor inputs. Crucially, most of these causes correspond to the action of the robot itself.

of legs appropriately for the current state, while the arms coordinate their joints through Inverse Kinematics (IK).

The low-level joint controller is instantiated as joint impedance control and tracks the joint position commands afforded by the mid-level stability controller. Based on tuned stiffness and damping, the joint impedance control calculates the desired torque to attain target positions closely and smoothly. Lastly, the torque commands are tracked by the actuators, using embedded current control of onboard motor drivers.

# 3. RESULTS

In the following, we illustrate how our proposed formulation of control architectures as inversion of a hierarchical generative model. Specifically, we show how this formulation enables a robot to learn how to complete a loco-manipulation task autonomously. Additionally, the learned policy features generality and robustness to uncertainty (Fig. 3 a-e), while evincing the core principles of hierarchical motor control. First, we validate the notion that hierarchical generative modelling is sufficient for autonomous robotic operations. We demonstrate this in two sequential decision-making tasks: moving a box from one table to another and opening a door by pressing a button.

Our implicit hierarchical generative model can successfully and autonomously achieve locomotion, manipulation, and grasping movements similar to humans, and solve all these complex tasks coherently and with internal consistency. Briefly, the highest policy level of the robot system determines the action sequence necessary for task completion and sends commands to the lower levels. This allows the robot to carry out the following actions: i) walk to the first table, ii) move arms to pick up the box, iii) walk to the second table, iv) and place the box on the table. Upon successful completion of this task, the same generative model can be used to perform the second task as well. This is achieved by using the high-level policy to open a closed door, instead of moving a box. The accompanying commands are sent to the lower levels responsible for limb stability and joint control. They instantiate mid-level locomotion

and manipulation policies to move to the door and position the body close enough to press and opening button. Having opened the door, the robot enters, and proceeds towards the final goal.

To assess the robustness and generality of this hierarchical scheme, we introduced several perturbations that were not encountered during training (Fig. 3). First, we introduced external perturbation by placing obstacles (i.e., 5kg box, Fig. 3a) in front of the robot and pushing its pelvis (Fig. 3b). The results were encouraging: the mid-level locomotion policy withstood both perturbations, moved the obstacle out of the way, and took a step to recover balance after the push. To test the performance further, we modified the environment with unseen conditions by adding a 5° inclined surface (Fig. 3c) and a low-friction glass plate (friction coefficient of 0.3, Fig. 3d) in front of the door. The robot could complete the task after each perturbation. More interestingly, we lesioned the robot by amputating its right foot (Fig. 3e). Despite this handicap (that was never encountered)—and with only a stump touching the ground in place of its right foot—our hierarchical control was sufficiently robust to deal with this situation and the robot was able to keep balance and complete the task.

Next, we evaluate whether the ensuing control architecture satisfies the key principles of hierarchical motor control (Table 1) that underwrite robust task performance.

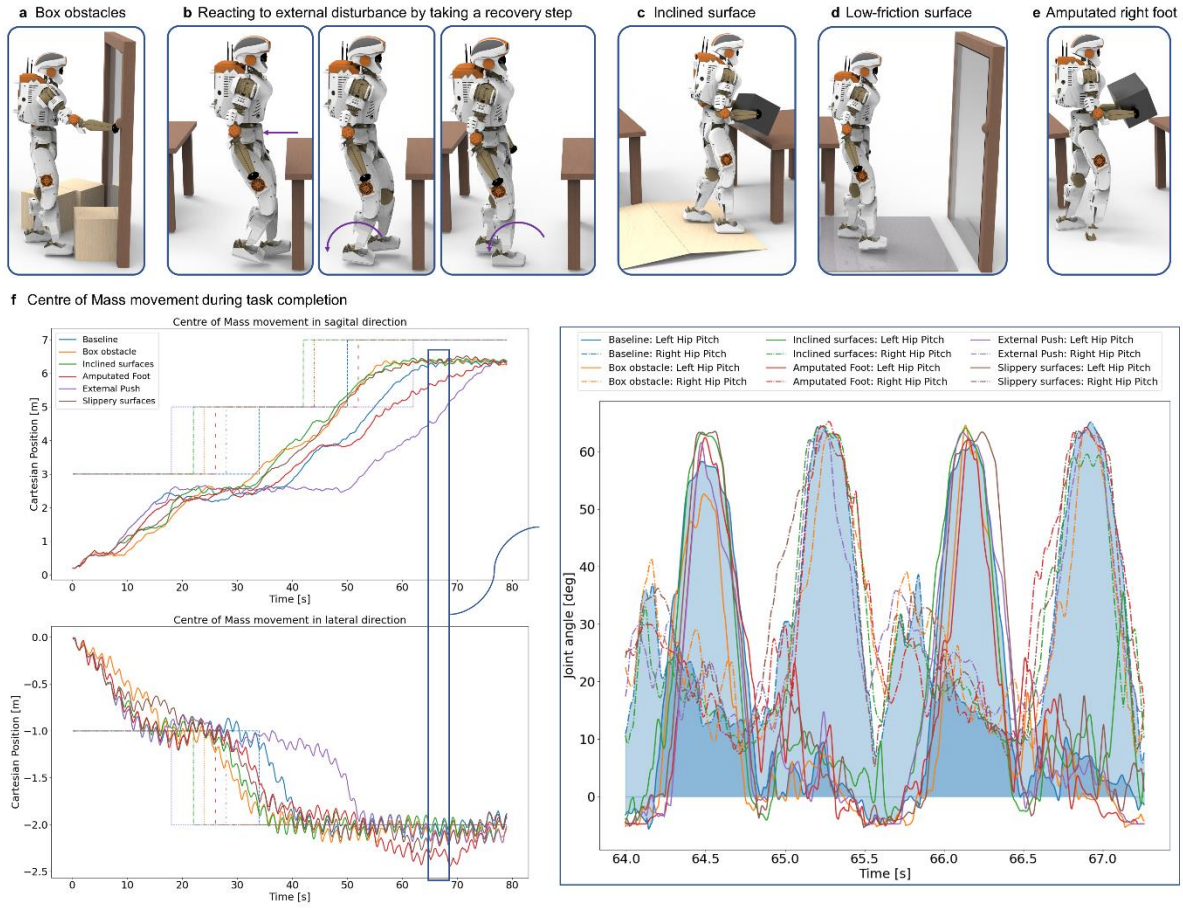


**Figure 3:** Robustness of the system in light of perturbations. Panels A-E show how the robot completes the task in perturbation test scenarios that it has not encountered during training and demonstrate the robustness of our proposed method. From left to right, we place 5kg box-obstacles in front of the robot, push the robot from the front, alter the floor with an inclined and slippery surface, and lesion the robot, but removing the right foot. In Panel F, the amortised control is shown. The hip pitch joint motion is used to show how the policy adapts to the perturbation and rapidly re-executes a motion to counteract the perturbation.

## INFORMATION FACTORISATION

In this system, factorisation exists across model levels and policy controls, each responsible for a particular sort of information processing. This factorisation ensures that external perturbations have minimum impact on task performance. For example, when disturbances involve robot legs, these perturbations do not significantly impede task completion. This is because the perturbations are only visible to the locomotion policy, and other levels and control policies are conditionally independent of the disturbances and hence retain their nominal operations.

Practically, information factorisation necessitates clearly defined roles for each sub-system. Thus, any failures in performance can be isolated and fine-tuned for future tasks. For example, if the robot falls

over while walking to a goal, the locomotion policy can be identified as the root cause, and hence improving the locomotion policy will resolve the issue without needing to modify the high-level planner or the manipulation policy. Further examples include oscillation of the robot limbs, which can be attributed to the low-level joint control; or walking in the wrong direction, which was due to the command from the high-level policy. From a theoretical perspective, factorisation of this sort corresponds to the structure of the generative model that can be decomposed into factors of a probability distribution. Almost universally, this results in certain conditional independencies that minimise the complexity of model inversion; namely, planning or control as inference [9,11,13,43]. This is important because it precludes over fitting and ensures generalisation. From a biological perspective, this kind of factorisation can be regarded as a functional segregation that is often associated with modular architectures and functional specialisation in the brain [44].

## PARTIAL AUTONOMY

The system is designed with partial autonomy, i.e., minimum interference or support from other levels. Specifically, we implement a clear separation between the highest and intermediate levels, though they are learned together. This is particularly relevant because the high-level planning level could send unrealisable action sequences to the mid-level stability controller. Without partial autonomy, the robot might become unstable and unable to learn to move appropriately, given such random or potentially unstable high-level commands.

Figure 4 illustrates this case when the robot is provided with random commands to both the arms and legs. This causes the robot to walk in random directions (Fig. 4a) and the arms move around randomly (Fig. 4b). Despite imperfect motion tracking, the robot does not fall over and can complete the tasks despite incoherent intentions.

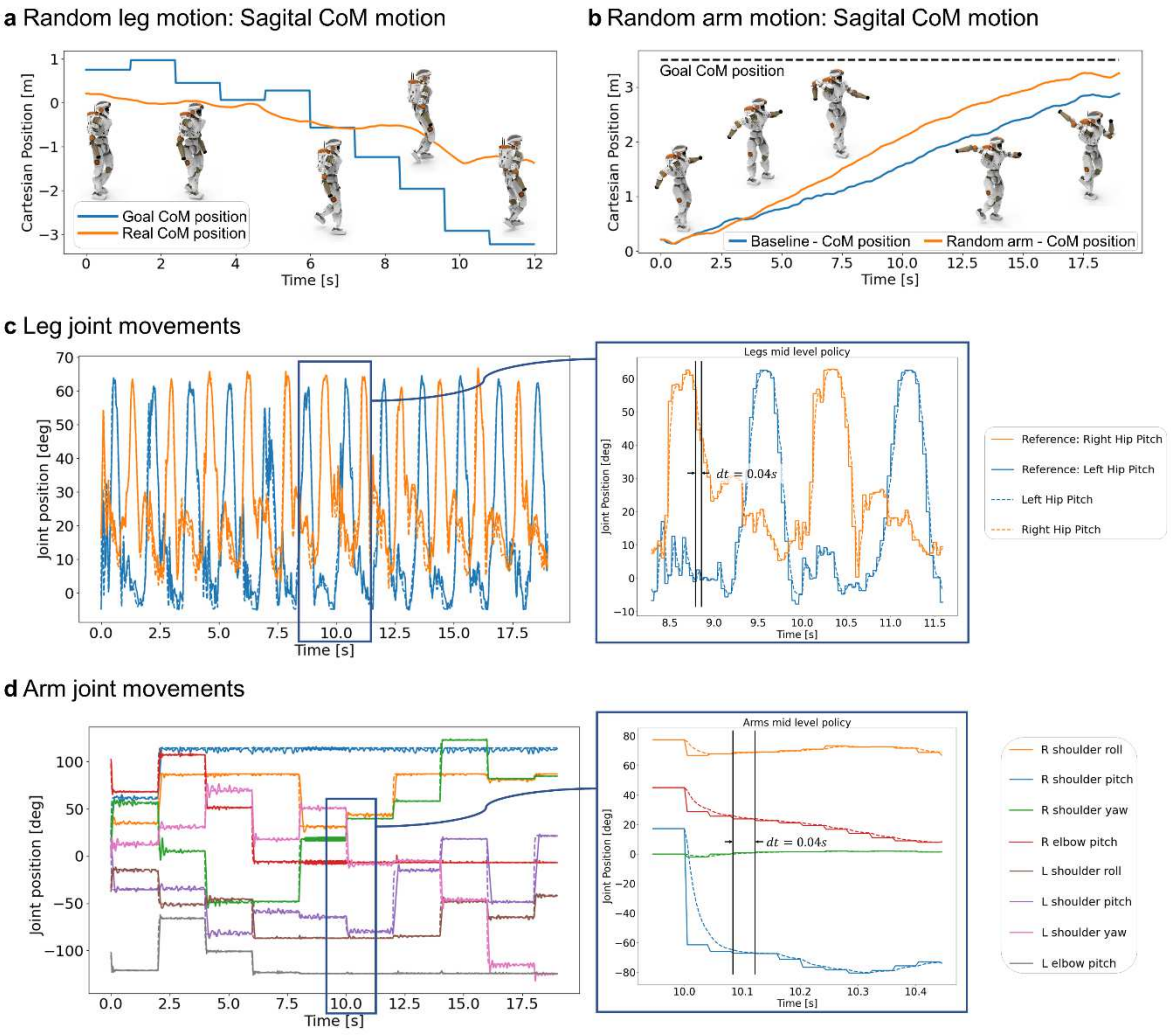


**Figure 4:** State and temporal dynamics of the robot during task performance with random-high level commands. Panels a and b show the sagittal motion of the Centre of Mass (CoM) while following random leg and arm commands respectively. From the robot snapshots corresponding to the time they're shown, the partial autonomy of the mid-level stability controllers can be seen, i.e., good performance of the individual levels despite random and fast-changing command inputs. Panels c and d show the leg and arm movements respectively. Here, the separation of temporal scales during planning can be seen, where the high-level commands are provided at 0.5 Hz, and the mid-level commands are realised at 25Hz. The joint commands are realised at 500Hz on the joint actuators. In the inset plots of Panels c and d, it can be seen that the joint position trajectories evolve similarly as postulated in the equilibrium point hypothesis.

## AMORTISED CONTROL

After training, the robot engages in amortised control with the ability to re-execute appropriate behaviours rapidly using previously learnt movements. We observed this in the baseline and perturbed task settings (inset trajectories in Fig. 3f), where the amortised locomotion policy was used to complete the task without additional learning.

## MULTI-JOINT COORDINATION

The robot has multiple sub-structures that are responsible for specific controls, and work together in different ways to generate particular motor movements. Figure 5a demonstrates this multi-joint coordination when pressing the button to open the door in the presence of an obstacle (Task 2). To achieve this, the right arm motions had to coordinate appropriately according to the initial hand position. Also, the shoulder roll (Fig. 5b orange line) and elbow (Fig. 5b red line) had to adjust and adapt differently from the baseline. Explicitly, these do not yield a fixed motion, instead, the manipulation policy coordinates these joints based on the Centre of Mass (CoM). Therefore, during the baseline reaching motion, the arms move differently than in the case of an obstructed box, where the CoM is in a different position (because boxes are obstructing the door).

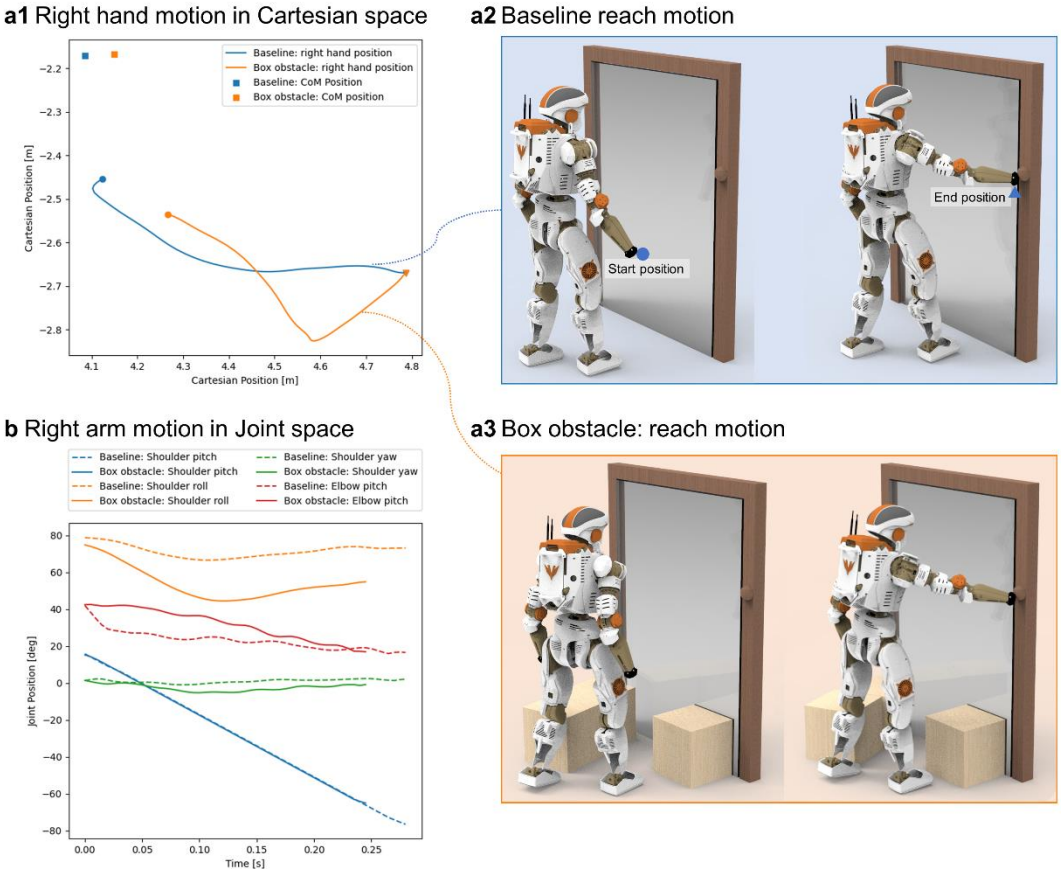


**Figure 5:** Adaptive arm manipulations during opening the door task. Panel a shows the right arm's motion in cartesian space for the baseline (no obstacle) and the box obstacle scenario. When there is an obstacle in front of the door, the arm motion must adapt, as the robot's upper body position deviates from its baseline position. The multi-joint coordination aspect can be seen in Panel b. The adapts to the change of scenario, coordinating the multiple arm joints, such that the end position of the arm is pressing the button to open the door, while being obstructed by box obstacles.

### TEMPORAL ABSTRACTION AND DEPTH

By design, the three system levels evolve at different temporal scales. Figure 4 illustrates these distinct scales as the robot perambulates. The highest policy level has a slow timescale of 0.5Hz (Fig. 4a). This allows the lower levels to carry out the command in a partially autonomous way, i.e., uninterrupted. Conversely, the mid-level stability control for limbs has a faster timescale at 25Hz (inset trajectories of Fig. 4c and 4d). This is needed to generate rapid predictions for the locomotion and manipulation

policies. Finally, the low-level joint control executes these control commands at a frequency of 500Hz on the actuator level.

# 4. DISCUSSION

We have shown that hierarchical generative models can adequately equip agents with the ability to perform autonomously, in a context sensitive and robust fashion. Hierarchical generative modelling provides sensorimotor control systems with a transparent, and flexible, approach to interpret (and implement) robotic decision making. Importantly, this can be leveraged to improve task performance and identify system failures. It is worth acknowledging that the ensuing hierarchical framework adheres to the key principles of hierarchical motor control, and therefore offers the possibilities to (i) create a flexible system, (ii) rollout the policy and evaluate its performance, (iii) identify the root cause of the limitations in performance, and (iv) resolve issues in the root cause and improve the performance.

The proposed hierarchical generative model illustrates the effectiveness of hierarchical motor control principles for designing fully autonomous and robust robotics systems. We validated this approach on robot loco-manipulation tasks that required the retrieval and delivery of a box, opening a door, and walking towards a final goal. Furthermore, we showed the robustness of our system by demonstrating successful task completion despite previously unencountered perturbations, such as novel changes to the environment, unanticipated pushes, and even amputation of the right foot.

By providing robots with full autonomy with appropriate triage procedures, humans can be relieved of having to send low-level commands every few seconds (e.g., foot and hand placement) that require the interpretation of a steady stream of percept and sensor data—as in a shared autonomy and semi-autonomous paradigm. Consequently, human errors induced by a cognitive overload may be prevented in such robotics systems. Previously, these human errors have been observed in high-stress situations, despite being operated by experienced individuals with high social and hardware costs. A notable

example is the robot crash caused by incorrect foot placement during the DARPA Robotics Challenge—the most advanced competition to test the capabilities of anthropomorphic disaster-response robots.

Future work could evaluate the implications of more nuanced planning objectives, and their implications for such autonomous robotics systems. We expect that replacing our Q-learning planner, with more sophisticated objectives equipped to handle aleatoric and epistemic uncertainties (e.g., expected free energy [45-47]), could improve the performance of the robot when dealing with volatile contingencies [46]. Furthermore, the robustness of our method could be evaluated through robotic neuropsychology [48], i.e., introducing in-silico lesions to the robot system, and investigating their effect on the resulting policies, inference and behaviour.

# 5. METHODS

Here, we present the hardware implementation for inverting the (implicit) hierarchical generative (a.k.a. forward) model for autonomous robot control. First, we describe the robot platform on which the algorithms are implemented. Then, we detail the task that is solved autonomously by inverting the generative model; i.e., using the model to predict sensor inputs and using actuators to resolve the ensuing (proprioceptive) prediction errors. Next, we explain the generative model details including high-level decision making, mid-level stability control, and low-level joint control.

## ROBOT PLATFORM

The motions for autonomous task completion are implemented on NASA's humanoid Valkyrie[49]. Valkyrie was designed to operate in extra-terrestrial planetary space missions such as unmanned pre-deployment on Mars – it is 1.87m tall, consists of 44 Degrees of Freedom, and weighs 129kg with ranges of motion similar to humans. The 25 series-elastic actuators in arms, torso, and legs enable human-like locomotion and manipulation; all the joint limits are detailed in Figure S1. Valkyrie can sense the environment through proprioceptive and exteroceptive sensors, including a multitude of

gyroscopes, accelerometers, load cells, pressure sensors, sonar, LIDAR, depth cameras, and stereo sensors.

## TASKS OF INTEREST

To demonstrate how in inversion of—or inference under—a hierarchical generative model solves complex tasks that require a particular sequence and coordination of locomotion and manipulation skills, we designed a task that demanded both coordination of limbs and reasoning about the sequence of actions. This task comprised four subtasks (Fig. S2): picking up a box from the first table, delivering the box to the second table, opening the door, and walking to the destination or goal position. To complete the task, all the subtasks had to be carried out in an exact sequence.

Our proposed framework allowed the robot to learn successful task completion through autonomous interactions with the environment. This was achieved by designing a reward (or utility) function for the high-level policy, such that cumulative maximisation of reward leads to task completion (see Section "High-level Decision Making").

The trained policy learnt to first pick up the box, and hold the box using its hands, while approaching the second table. Once the box could be safely placed at the second table, the selected policy ensured the box was released and the robot moved towards the door, while keeping its arms in an appropriate position. Next, by pushing the button placed on the right side of the door, the robot could open the door. The learnt policy ensured that the final destination was approached at the right time, i.e., when the door was sufficiently open to walk through—towards the goal position.

## IMPLICIT GENERATIVE MODEL

The generative model was realised by implementing three levels of control in a hierarchical manner (Fig. 2 in the manuscript): high-level decision making, mid-level stability control, and low-level joint control. All components were designed and trained separately, starting from the lowest level.

First, accurate and robust motor control needed to be guaranteed, such that the low-level joint position control could be realised. Stiffness and damping parameters were tuned to track the references accurately and compliantly, which provided the mid-level stability control. The mid-level stability control consisted of a manipulation and a locomotion policy, which were individually designed. The locomotion policy was trained to walk towards a commanded goal position, while the manipulation policy was designed to place the hands on a target position. Finally, the high-level decision-making policy was trained via Deep Reinforcement Learning, which learnt to provide appropriate commands to these mid- and low-level policies.

**High-level Decision Making**

We achieved high-level decision making, the correct sequence and choices of robot actions, through training a Deep Neural Network that approximated the action value function $Q(s,a)$ over the environment and choose the action $a$ which yielded the highest value in state $s$.

***Double Q-learning***

We used Double Q-learning [50] to train a Q-network $Q(s,a;\phi)$, parametrised by weights $\phi$, to approximate the true action value function $Q(s,a)$. At run-time, the action $a$ was obtained as the argument of the maximum Q-value $a = \text{argmax}_a Q(s,a;\phi)$ in state $s$. Compared to Deep Q-Learning[51], we used two separate Q-networks $Q_1, Q_2$ for action selection and value estimation. Having two separate Q-networks has previously shown to improve training stability[50].

The network parameter $\phi_i$ was obtained by $\min_{\phi_i} L(\phi_i)$:

$$\min_{\phi_i} E\left[\left(r + \gamma Q_j\left(s', a^*; \phi_j\right) - Q_i(s,a;\phi_i)\right)^2\right],$$

with reward $r$, discount factor $\gamma$, network parameters $\phi_i, \phi_j$, Q-networks $Q_i, Q_j$, current state $s$, next state $s'$, best action $a^* = \text{argmax} Q_i(s,a;\phi_i)$. During training, either network parameters $\phi_1$ or $\phi_2$ was randomly selected, trained, and used for action selection, while the other network parameter was used

to estimate the action value. The tuple $(s, a, r, s') \sim U(D)$ was obtained from the Experience Replay by uniformly sampling from buffer $D$, which was updated by online action rollout.

***Training Procedures***

The high-level policy sent and updated the actions $a \in \mathcal{A} \subseteq \mathcal{R}^9$ at $1Hz$ frequency, which were the positions in Cartesian space for the pelvis $a_{\text{pelvis}} \in \mathcal{R}^3$, left and right hands $a_{\text{lh}}, a_{\text{rh}} \in \mathcal{R}^3$. These actions $a$ were executed by the mid-level stability controller. The locomotion policy constituted walking towards the pelvis targets $p_{\text{Pelvis}}$, while the manipulation policy placed the hands towards the hand targets $p_{\text{lhand}}, p_{\text{rhand}}$.

The states $s \in \mathcal{S} \subseteq \mathcal{R}^9$ were the vector $\vec{s}_{\text{pelvis}} = a_{\text{pelvis}} - p_{\text{Pelvis}} \in \mathcal{R}^3$ from current pelvis position $p_{\text{Pelvis}}$ to the pelvis target $a_{\text{pelvis}}$, the vector $\vec{s}_{\text{hands}} = a_{\text{lh, rh}} - p_{\text{lh, rh}} \in \mathcal{R}^6$ from current hand positions $p_{\text{lh, rh}}$ to left and right hand targets $a_{\text{lh, rh}}$. Lastly, three Boolean variables were provided as the observation state when the door was open, the box was on the table, or the box was being carried.

The reward terms $r_i$ were determined based on the task completion, such as whether the robot had passed the delivery table, the arm joints were in the nominal position, the box was between the robot hands, the box was at the delivery table, the door was open, and whether the robot was at the goal. The weights $w_i, i = 1, \ldots 6$ can be found in Figure S3.

At every time step, the reward $r$ was the sum of sparse, Boolean states:

$$r = w_1 r_{\text{pt}} \quad + w_2 r_{\text{jn}} + w_3 r_{\text{bih}} + w_4 r_{\text{bot}} + w_5 r_{\text{do}} + w_6 r_{\text{ag}},$$

with passed table reward $r_{\text{pt}}$, joints nominal reward $r_{\text{pt}}$, box in hand reward $r_{\text{pt}}$, box on table reward $r_{\text{pt}}$, door open reward $r_{\text{pt}}$, and at goal reward $r_{\text{pt}}$.

We terminated the episode early if the robot fell, or collided with itself, tables, or the door. Early termination discouraged the policy from entering sub-optimal states (and actions) that would lead to early termination as the cumulative reward becomes low, due to reaching the end of an episode.

We initialised the robot in different positions of the environment to allow the robot to encounter states that were hard to discover merely by exploration. Additionally, the robot was deliberately spawned at the configurations that were close to the final goal, in front of the open or closed door, and front of the tables.

**Mid-level Stability Control**

The mid-level stability control level consisted of two components: the manipulation policy for the arms realised as Model-Predictive Control (MPC) scheme, and a locomotion policy for the legs learned through Deep Reinforcement Learning.

***Manipulation Policy***

The manipulation policy received Cartesian target positions for the hands from the High-level Decision-Making policy and sent joint position information to the Low-level Joint Controller. This was achieved by combining two parts (Fig. S4): Model-Predictive Control (MPC) that generated a stable, optimal trajectory in Cartesian space and Inverse Kinematics (IK) [52] that transformed desired actions from the Cartesian space to the joint space.

To provide the smoothest possible motions for the hands, we formulated the optimal control problem as the minimum-jerk optimisation, while satisfying dynamics constraints on the hands. The optimal trajectory was then implemented in an MPC fashion. The MPC control applied the first control input of the optimal input trajectory and then re-optimised based on the new state at the next control loop[53]. In this way, MPC successively solved an optimal control problem over a prediction horizon $N$ and achieved feedback control, while ensuring optimality.

For the hand position $x$, an objective function $J$ was designed to minimise jerk $\dddot{x}$ (the input $u$ of the system) with final time $t_f$:

$$J = \frac{1}{2}\int_0^{t_f}\left(\frac{\mathrm{d}^3 x(t)}{\mathrm{d}t^3}\right)^2 \mathrm{d}t = \frac{1}{2}\int_0^{t_f} u\,(t)^2 \mathrm{d}t.$$

The Minimum Jerk MPC (MJMPC) solved the following constrained optimisation problem at very time step at a frequency of $25Hz$:

$$
\begin{aligned}
2 \quad & \min_{u(t)} && \frac{1}{2}\int_0^{t_f} u\,(t)^2 \mathrm{d}t \\
& \text{subject to} && \frac{\mathrm{d}^3 x(t)}{\mathrm{d}t^3} = u \\
& && [x(0), \dot{x}(0), \ddot{x}(0)] = [x_0, \dot{x}_0, \ddot{x}_0] \\
& && \left[x(t_f), \dot{x}(t_f), \ddot{x}(t_f)\right] = \left[x_f, \dot{x}_f, \ddot{x}_f\right] \\
& [x_{\min}, \dot{x}_{\min}, \ddot{x}_{\min}] && \leq [x, \dot{x}, \ddot{x}] \leq [x_{\max}, \dot{x}_{\max}, \ddot{x}_{\max}],
\end{aligned}
$$

with initial condition $[x_0, \dot{x}_0, \ddot{x}_0]$, and terminal condition $\left[x_f, \dot{x}_f, \ddot{x}_f\right]$.

The resultant Cartesian trajectory $p^d$ from MJMPC was transformed into joint position commands $\theta^d$ through Inverse Kinematics (IK). More formally, IK described a transformation $T: \mathcal{C} \to \mathcal{Q}$ from Cartesian space $\mathcal{C}$ to joint space $\mathcal{Q}$. The joint position commands $\theta^d$ were then tracked by the low-level joint position controller as described in Section "Low-level Joint Control".

***Locomotion Policy***

The locomotion policy $\pi(s;\theta)$ coordinated the 12 Degree of Freedom (DoF) leg joints and was instantiated as a Deep Neural Network (network parameters $\theta$) that received robot states $s$ as inputs and outputs 12 target joint positions for legs. It was trained through Soft-Actor Critic (SAC) [54]– an off-policy Deep Reinforcement Learning (DRL) algorithm.

SAC optimised a maximum entropy objective $J_{\mathrm{SAC}(\pi)}$:

$$J_{\mathrm{SAC}}(\pi) = \sum_{t=0}^{T} \mathbb{E}\left[r(s_t, a_t) + \alpha \mathcal{H}\left(\pi(\cdot\,|s_t)\right)\right],$$

with reward $r$, state $s_t$ and action $a_t$ at time $t$, temperature parameter $\alpha$, and policy entropy $\mathcal{H}(\pi)$.The parameters $\theta$ for policy $\pi_\theta$ were obtained by minimising $J_\pi(\theta)$:

$$J_\pi(\theta) = \mathbb{E}\left[\log \pi_\theta(a_t|s_t) - Q_\phi(s_t, a_t)\right].$$

The action-value function $Q_\phi(s_t, a_t)$ was obtained by minimising the Bellman residual $J_Q(\phi)$:

$$J_Q(\phi) = \mathbb{E}\left[1/2\left(Q_\phi(s_t, a_t) - \hat{Q}(s_t, a_t)\right)^2\right],$$

with Bellman equation $\hat{Q}(s_t, a_t) = r(s_t, a_t) + \gamma\mathbb{E}\left[V_\psi(s_{t+1})\right]$ and discount factor $\gamma$. Estimation of the value function $V_\psi$ was obtained through minimising $J_V(\psi)$:

$$J_V(\psi) = \mathbb{E}[1/2\left(V_\psi(s_t) - \mathbb{E}\left[Q_\theta(s_t, a_t) - \log\pi_\phi(a_t|s_t)\right]\right)^2].$$

The training process of policy $\pi_\theta$ is detailed in Algorithm 1.

**Algorithm 1** Pseudocode for SAC

$\mathcal{D} \leftarrow \emptyset$ , initialise replay buffer
**for** iter=1,2,... **do**
  **while** collected samples < batch size **do**
    Sample and perform action $a_t \sim \pi_\theta$
    Collect $d = \{s_t, a_t, r(s_t, a_t), s_{t+1}\}$
    Store sample $d$ in replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup d$
  **for** policy update=1,...,K **do**
    Sample batch from replay buffer $\mathcal{D}$ for update
    Update policy $\pi_\theta$ via (1.5)
    Update action value function $Q_\phi$ via (1.6)
    Update value function $V_\psi$ via (1.7)

The training procedures, including the design of reward, action space, and state-space, are as in[55]. The action space $\mathcal{A} \in \mathcal{R}^{12}$ were the joint positions of the 12 Degree of Freedoms (DoF) of the legs (for each leg 3 DoF hip, 1 DoF knee, 2 DoF ankle). The target joint positions were tracked by the low-level joint controller (see Section "Low-level Joint Control").

The state-space $\mathcal{S} \in \mathcal{R}^{27}$ consisted of the desired pelvis position/reference (the walking destination), and proprioceptive feedback states of the robot including pelvis orientation, linear and angular velocity of the pelvis, force of both feet, joint positions of the legs, and the gait phase.

The reward comprised of an imitation term and a task term similar to :

$$r_i = w_i r_{\text{imitation}} + w_t r_{\text{task}},$$

with weights $w_i, w_t$ and reward terms $r_{\text{imitation}}, r_{\text{task}}$ for imitation and task respectively.

The aim of $r_{\text{imitation}}$ was to imitate the joint position, feet pose, and contact pattern of a reference motion capture trajectory as closely as possible. The reward term $r_{\text{task}}$ rewarded upright posture, short distances to the goal position, and regularised the joint velocity and torque.

**Low-level Joint Control**

The low-level joint control tracked the target joint positions $\theta^d$ provided by the mid-level stability controller. It was realised as *Joint Impedance Controller* that regulated around the set point. Additionally, a feed-forward controller (*Inverse Dynamics* block in Fig. S5) provided joint torques $\tau^{FF}$ that compensated for the dynamical influences from gravity and Coriolis forces and hence achieved more accurate tracking of the desired joint motions $\theta^d$.

***Feedback Control***

The desired control effort was calculated using stiffness $K_{P_i}$ and damping gains $K_{D_i}$. The joint impedance control calculated the desired joint torque $\tau^d$ using position $\theta$ and its derivative $\dot{\theta}$:

$$\tau^d = K_{P_1}\left(\theta^d - \theta\right) - K_{D_1}\dot{\theta}.$$

At the actuator level, the motor driver implemented an internal current control to track the desired joint torque $\tau^d$ using a proportional-derivative law, where the desired motor current $I$ was computed as:

$$I = K_{P_2}\left(\tau^d - \tau\right) - K_{D_2}\dot{\tau}.$$

***Feedforward Control***

The feedforward torques $\tau^{FF}$ were computed through Inverse Dynamics, which transformed $T: \Theta \rightarrow \mathrm{T}$ from desired joint motions $\theta^d \in \Theta$ to joint torques $\tau^{FF} \in \mathrm{T}$, based on the equations of motions satisfying the Newton–Euler dynamics of multi-link rigid bodies given the joint configuration $\theta^d, \ddot{\theta}^d$. These resultant torques $\tau^{FF}$ compensated for gravity and Coriolis forces given the desired joint configuration $\theta^d$.

# REFERENCES


1 Li, N., Chen, T.-W., Guo, Z. V., Gerfen, C. R. & Svoboda, K. A motor cortex circuit for motor planning and movement. *Nature* **519**, 51-56, doi:10.1038/nature14178 (2015).

2 Rosenbaum, D. A., Vaughan, J., Barnes, H. J. & Jorgensen, M. J. Time course of movement planning: Selection of handgrips for object manipulation. *Journal of Experimental Psychology: Learning, Memory, and Cognition* **18**, 1058-1073, doi:10.1037/0278-7393.18.5.1058 (1992).

3 Svoboda, K. & Li, N. Neural mechanisms of movement planning: motor cortex and beyond. *Current opinion in neurobiology* **49**, 33-41 (2018).

4 Hogan, N. Planning and execution of multijoint movements. *Canadian Journal of Physiology and Pharmacology* **66**, 508-517 (1988).

5 Honey, C. J. *et al.* Slow cortical dynamics and the accumulation of information over long timescales. *Neuron* **76**, 423-434, doi:10.1016/j.neuron.2012.08.011 (2012).

6 Murray, J. D. *et al.* A hierarchy of intrinsic timescales across primate cortex. *Nat Neurosci* **17**, 1661-1663, doi:10.1038/nn.3862 (2014).

7 Diedrichsen, J., Shadmehr, R. & Ivry, R. B. The coordination of movement: optimal feedback control and beyond. *Trends in cognitive sciences* **14**, 31-39, doi:10.1016/j.tics.2009.11.004 (2010).

8 Johnson, M. *et al.* Team IHMC's Lessons Learned from the DARPA Robotics Challenge Trials. *Journal of Field Robotics* **32**, 192-208 (2015).

9 Attias, H. in *Proc. of the 9th Int. Workshop on Artificial Intelligence and Statistics.*

10 Baker, C. L., Saxe, R. & Tenenbaum, J. B. Action understanding as inverse planning. *Cognition* **113**, 329-349, doi:10.1016/j.cognition.2009.07.005 (2009).

11 Botvinick, M. & Toussaint, M. Planning as inference. *Trends Cogn Sci.* **16**, 485-488 (2012).

12 Maisto, D., Donnarumma, F. & Pezzulo, G. Divide et impera: subgoaling reduces the complexity of probabilistic inference and problem solving. **12**, 20141335, doi:10.1098/rsif.2014.1335 (2015).

13 Kaplan, R. & Friston, K. J. Planning and navigation as active inference. *Biological cybernetics* **112**, 323-343, doi:10.1007/s00422-018-0753-2 (2018).

14 Tani, J. & Nolfi, S. Learning to perceive the world as articulated: an approach for hierarchical learning in sensory–motor systems. *Neural Netw* **12**, 1131–1141 (1999).

15 Tani, J. Learning to generate articulated behavior through the bottom-up and the top-down interaction processes. *Neural Netw.* **16**, 11-23 (2003).

16 Jung, M., Hwang, J. & Tani, J. Self-Organization of Spatio-Temporal Hierarchy via Learning of Dynamic Visual Image Patterns on Action Sequences. *PLOS ONE* **10**, e0131214, doi:10.1371/journal.pone.0131214 (2015).

17 Matsumoto, T. & Tani, J. Goal-Directed Planning for Habituated Agents by Active Inference Using a Variational Recurrent Neural Network. *Entropy* **22**, 564, doi:10.3390/e22050564 (2020).

18 Haruno, M., Wolpert, D. M. & Kawato, M. Hierarchical MOSAIC for movement generation. *International Congress Series* **1250**, 575-590, doi:https://doi.org/10.1016/S0531-5131(03)00190-0 (2003).

19 Wolpert, D. M., Doya, K. & Kawato, M. A unifying computational framework for motor control and social interaction. *Philos Trans R Soc Lond B Biol Sci.* **358**, 593-602 (2003).

20 Morimoto, J. & Doya, K. Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems* **36**, 37–51 (2001).

21 Doya, K. *Bayesian brain: Probabilistic approaches to neural coding*. (MIT press, 2007).

22 Baltieri, M. & Buckley, C. L. Generative models as parsimonious descriptions of sensorimotor loops. *Behavioral and Brain Sciences* **42**, e218, doi:10.1017/S0140525X19001353 (2019).

23 Friston, K. J., Parr, T. & de Vries, B. The graphical brain: Belief propagation and active inference. *Network neuroscience* **1**, 381-414, doi:10.1162/NETN_a_00018 (2017).

24 Merel, J., Botvinick, M. & Wayne, G. Hierarchical motor control in mammals and machines. *Nature Communications* **10**, 5489, doi:10.1038/s41467-019-13239-6 (2019).

25 Parr, T., Limanowski, J., Rawji, V. & Friston, K. The computational neurology of movement under active inference. *Brain : a journal of neurology* (2021).

26 Jackson, J. H. On certain relations of the cerebrum and cerebellum (on rigidity of hemiplegia and on paralysis agitans). *Brain : a journal of neurology* **22**, 621-630 (1899).

27 Pezzulo, G., Rigoli, F. & Friston, K. J. Hierarchical Active Inference: A Theory of Motivated Control. *Trends in cognitive sciences* **22**, 294-306, doi:10.1016/j.tics.2018.01.009 (2018).

28 Feldman, A. G. & Levin, M. F. The equilibrium-point hypothesis--past, present and future. *Advances in experimental medicine and biology* **629**, 699-726, doi:10.1007/978-0-387-77064-2_38 (2009).

29 Perrier, P., Ostry, D. J. & Laboissière, R. The Equilibrium Point Hypothesis and Its Application to Speech Motor Control. *Journal of Speech, Language, and Hearing Research* **39**, 365-378, doi:doi:10.1044/jshr.3902.365 (1996).

30 Botvinick, M. M., Niv, Y. & Barto, A. C. Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective. *Cognition* **113**, 262-280, doi:10.1016/j.cognition.2008.08.011 (2009).

31 Aitchison, L. & Lengyel, M. With or without you: predictive coding and Bayesian inference in the brain. *Current opinion in neurobiology* **46**, 219-227 (2017).

32 Adams, R. A., Shipp, S. & Friston, K. J. Predictions not commands: active inference in the motor system. *Brain Struct Funct.* **218**, 611-643 (2013).

33 Bizzi, E., Mussa-Ivaldi, F. & Giszter, S. Computations underlying the execution of movement: a biological perspective. *Science* **253**, 287-291, doi:10.1126/science.1857964 (1991).

34 Marder, E. & Bucher, D. Central pattern generators and the control of rhythmic movements. *Current biology : CB* **11**, R986-996, doi:10.1016/s0960-9822(01)00581-4 (2001).

35 Miall, R. C., Weir, D. J., Wolpert, D. M. & Stein, J. F. Is the cerebellum a smith predictor? *J Mot Behav.* **25**, 203-216 (1993).

36 Doya, K. What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? *Neural Netw* **12**, 961-974, doi:10.1016/s0893-6080(99)00046-5 (1999).

37 Freeman, J. H. & Steinmetz, A. B. Neural circuitry and plasticity mechanisms underlying delay eyeblink conditioning. *Learning & memory (Cold Spring Harbor, N.Y.)* **18**, 666-677, doi:10.1101/lm.2023011 (2011).

38 Koziol, L. F. *et al.* Consensus paper: the cerebellum's role in movement and cognition. *Cerebellum (London, England)* **13**, 151-177, doi:10.1007/s12311-013-0511-x (2014).

39 Ramnani, N. Automatic and controlled processing in the corticocerebellar system. *Prog Brain Res* **210**, 255-285, doi:10.1016/b978-0-444-63356-9.00010-8 (2014).

40 Grillner, S., Wallén, P., Saitoh, K., Kozlov, A. & Robertson, B. Neural bases of goal-directed locomotion in vertebrates—an overview. *Brain research reviews* **57**, 2-12 (2008).

41 Jueptner, M., Frith, C., Brooks, D., Frackowiak, R. & Passingham, R. Anatomy of motor learning. II. Subcortical structures and learning by trial and error. *Journal of neurophysiology* **77**, 1325-1337 (1997).

42 Sommer, M. A. The role of the thalamus in motor control. *Current Opinion in Neurobiology* **13**, 663-670, doi:https://doi.org/10.1016/j.conb.2003.10.014 (2003).

43 Millidge, B. Deep Active Inference as Variational Policy Gradients. *arXiv e-prints*, arXiv:1907.03876 (2019).

44 Parr, T., Sajid, N. & Friston, K. J. Modules or Mean-Fields? *Entropy* **22**, 552 (2020).

45 Friston, K., FitzGerald, T., Rigoli, F., Schwartenbeck, P. & Pezzulo, G. Active Inference: A Process Theory. *Neural Comput* **29**, 1-49, doi:10.1162/NECO_a_00912 (2017).

46 Sajid, N., Ball, P. J., Parr, T. & Friston, K. J. Active inference: demystified and compared. *Neural computation* **33**, 674-712 (2021).

47 Da Costa, L. *et al.* Active inference on discrete state-spaces: A synthesis. *Journal of Mathematical Psychology* **99**, 102447, doi:https://doi.org/10.1016/j.jmp.2020.102447 (2020).

48 Sajid, N. *et al.* Simulating lesion-dependent functional recovery mechanisms. *Scientific Reports* **11**, 7475, doi:10.1038/s41598-021-87005-4 (2021).

49 Radford, N. A. *et al.* Valkyrie: Nasa's first bipedal humanoid robot. *Journal of Field Robotics* **32**, 397–419 (2015).

50 Hasselt, H. Double Q-learning. *Advances in neural information processing systems* **23**, 2613–2621 (2010).
51 Mnih, V. *et al.* Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).
52 Siciliano, B., Khatib, O. & Kröger, T. *Springer handbook of robotics*. Vol. 200 (Springer, 2008).
53 Rawlings, J. B., Mayne, D. Q. & Diehl, M. *Model predictive control: theory, computation, and design*. Vol. 2 (Nob Hill Publishing Madison, WI, 2017).
54 Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. in *International conference on machine learning*. 1861–1870.
55 Yang, C., Yuan, K., Heng, S., Komura, T. & Li, Z. Learning natural locomotion behaviors for humanoid robots using human bias. *IEEE Robotics and Automation Letters* **5**, 2610–2617 (2020).

## AUTHOR CONTRIBUTIONS

K.Y. and Z.L. conceptualised the robot control architecture. K.Y., N.S., and K.F. designed the hierarchical generative model. K.Y. implemented the model and performed the robotic experiments. K.Y. and N.S. wrote the manuscript. All authors contributed to and edited the manuscript.

## COMPETING INTERESTS

The authors declare no competing interests.

# Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- supplementary.pdf