ARTIFICIAL INTELLIGENCE

See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion

N. Fazeli[1]*, M. Oller[1], J. Wu[2], Z. Wu[2], J. B. Tenenbaum[2], A. Rodriguez[1]

Humans are able to seamlessly integrate tactile and visual stimuli with their intuitions to explore and execute complex manipulation skills. They not only see but also feel their actions. Most current robotic learning methodologies exploit recent progress in computer vision and deep learning to acquire data-hungry pixel-to-action policies. These methodologies do not exploit intuitive latent structure in physics or tactile signatures. Tactile reasoning is omnipresent in the animal kingdom, yet it is underdeveloped in robotic manipulation. Tactile stimuli are only acquired through invasive interaction, and interpretation of the data stream together with visual stimuli is challenging. Here, we propose a methodology to emulate hierarchical reasoning and multisensory fusion in a robot that learns to play Jenga, a complex game that requires physical interaction to be played effectively. The game mechanics were formulated as a generative process using a temporal hierarchical Bayesian model, with representations for both behavioral archetypes and noisy block states. This model captured descriptive latent structures, and the robot learned probabilistic models of these relationships in force and visual domains through a short exploration phase. Once learned, the robot used this representation to infer block behavior patterns and states as it played the game. Using its inferred beliefs, the robot adjusted its behavior with respect to both its current actions and its game strategy, similar to the way humans play the game. We evaluated the performance of the approach against three standard baselines and show its fidelity on a real-world implementation of the game.

INTRODUCTION

Humans, even young children, learn complex tasks in the physical world by engaging richly with the physics of the world (*1*). This physical engagement starts with our perception and extends into how we learn, perform, and plan actions. We seamlessly integrate touch and sight in coming to understand object properties and relations, allowing us to intuit structure in the physical world and to build physics representations that are central to our ability to efficiently plan and execute manipulation skills (*2*).

While learning contact-rich manipulation skills, we face two important challenges: active perception and hybrid behavior. In the former, we ask how we use temporal tactile and visual information gathered by probing our environment through touch to learn about the world. In the latter, we ask how we effectively infer and learn multimodal behavior to control touch. These two challenges are central to mastering physical interactions. Jenga is a quintessential example of a contact-rich task where we need to interact with the tower to learn and to infer block mechanics and multimodal behavior by combining touch and sight.

Current learning methodologies struggle with these challenges and have not exploited physics nearly as richly as we believe that humans do. Most robotic learning systems still use purely visual data, without a sense of touch; this fundamentally limits how quickly and flexibly a robot can learn about the world. Learning algorithms that build on model-free reinforcement learning (RL) methods have little to no ability to exploit knowledge about the physics of objects and actions. Even the methods using model-based RL or imitation learning have mostly used generic statistical models that do not explicitly represent the knowledge about physical objects, contacts, or forces that humans have from a very early age. As a consequence, these systems require far more training data than humans do to learn new models or new tasks, and they generalize much less broadly and less robustly.

In this work, we propose a hierarchical learning approach to acquiring manipulation skills. In particular, we pose a top-down bottom-up (*3*–*6*) learning approach to first build abstractions in the joint space of touch and vision that are then used to learn rich physics models. We used Jenga as a platform to compare and evaluate our approach. We have developed a simulation environment in which we compare the performance of our approach to three other state-of-the-art learning paradigms. We further show the efficacy of the approach on an experimental implementation of the game.

Our proposed approach draws from the notion of an "intuitive physics engine" in the brain that may be the cause of our abilities to integrate multiple sensory channels, plan complex actions (*7*–*9*), and learn abstract latent structure (*10*) through physical interaction, even from an early age. Humans learn to play Jenga through physics-based integration of sight and touch: Vision provides information about the location of the tower and current block arrangements but not about block interactions. The interactions are dependent on minute geometric differences between blocks that are imperceptible to the human eye. Humans gain information by touching the blocks and combining tactile and visual senses to make inferences about their interactions. Coarse high-level abstractions such as "will a block move" play a central role in our decision-making and are possible precisely because we have rich physics-based representations. We emulated this hierarchical learning and inference in the robotic system depicted in Fig. 1A using the artificial intelligence architecture schematically shown in Fig. 1B. To learn the mechanics of the game, the robot builds its physics-based representation from the data it collects during a brief exploration phase. We show that the robot builds purposeful abstractions that yield sample-efficient learning of a physics model of the game that it leverages to reason, infer, and act while playing the game.

[1]Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. [2]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.
*Corresponding author. Email: nfazeli@mit.edu

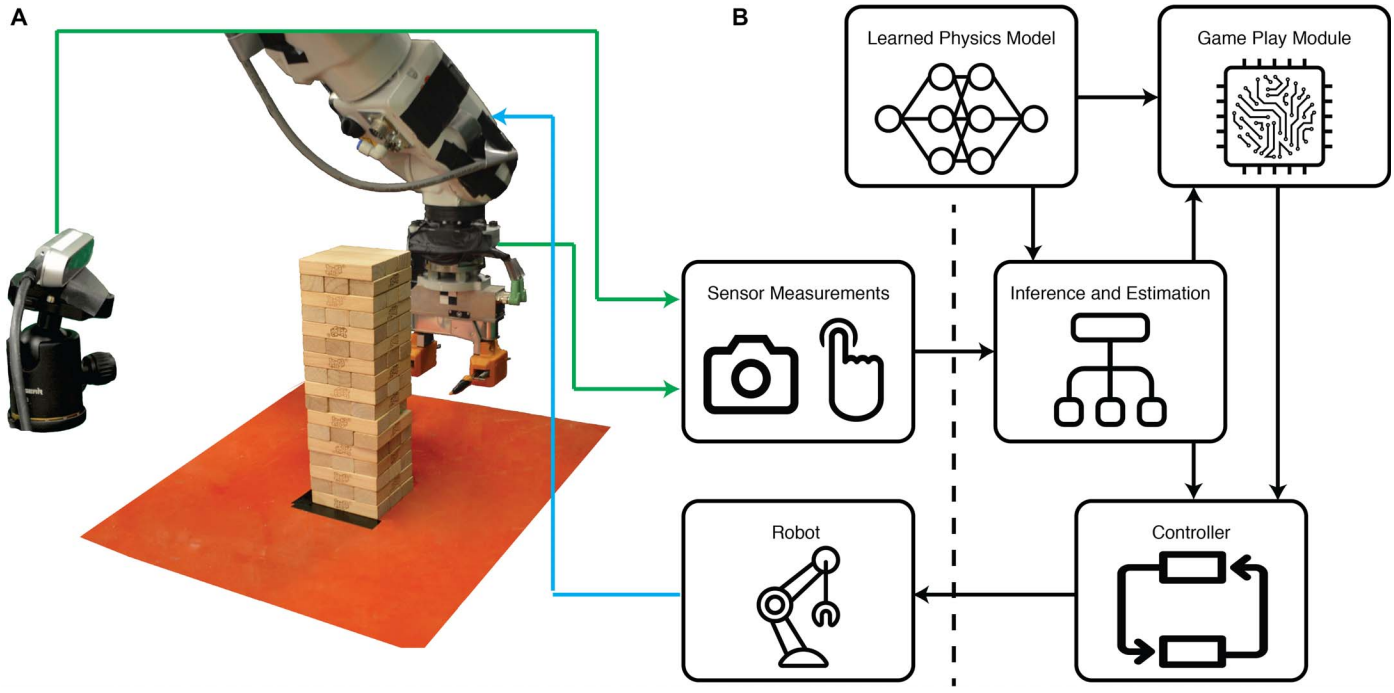

Fig. 1. Robot setup. (**A**) Physical setup consisting of the robot, Jenga tower, Intel RealSense D415 camera, and ATI Gamma force/torque sensor (mounted at the wrist). (**B**) Machine intelligence architecture with the learned physics model.

RESULTS

In this study, we demonstrate the efficacy and sample efficiency of a hierarchical learning approach to manipulation on the challenging game of Jenga. To this end, we first outline the task and evaluation metric. Next, we present quantitative results for our method and three competitive state-of-the-art alternative approaches in simulation. We then demonstrate the fidelity of our approach on a real-world implementation of the game. We end the section with an analysis of the physics and abstractions learned using the proposed approach.

Evaluation metric

Jenga is a particularly interesting instance of physical manipulation games, where mastering contact interactions is key to game play. Block mechanics are not discernible from just perception; rather, intrusive interaction, together with tactile and visual feedback, is required to infer and reason about the underlying latent structure, i.e., the different types of block mechanics.

The complex, partially observable, and multimodal mechanics of the blocks in Jenga pose a daunting challenge to robots learning to play the game. These challenges are not exclusive to this game and exist in many manipulation skills, such as assembly and tool use. Hence, progress in effective learning of these skills is important and necessitates rich models and policies.

In this study, we evaluated the robot's ability to play the game by counting the number of successful consecutive block extractions in randomly generated towers. This metric evaluates a model and/or policy's ability to account for the complex and time-varying mechanics of the tower as the game progresses. This metric, and our study, emphasizes physics modeling and does not explicitly evaluate the adversarial nature of the game.

Task specifications

In this subsection, we specify the sensing modalities, actions, and rules by which the robot is allowed to play the game in both simulated and real environments:

1) Sensing. The robot has access to its own pose, the pose of the blocks, and the forces applied to it at every time step. The simulated robot observes these states directly, whereas the experimental robot has access to noisy estimates.

2) Action primitives. The robot uses two "primitive" actions, push and extract/place. Using the push primitive, the robot first selects a block and moves to a collision-free configuration in plane. The robot then selects a contact location and heading and pushes for a distance of 1 mm and repeats. The action is considered complete if either the robot chooses to retract or a maximum distance of 45 mm is reached. The extract/place primitive searches for a collision-free grasp of the block and places it on top of the tower at a random unoccupied slot. The extract/place primitives are parametric and computed per call; hence, they are not learned.

3) Base exploration policy. The robot has access to a base exploration policy for data collection. This policy randomizes the push primitive by first selecting a block at random, then executing a sequence of randomized contact locations and headings.

4) Termination criteria. A run, defined as an attempt at a new tower, is terminated when one of the following conditions is met: (i) All blocks have been explored, (ii) a block is dropped outside the tower, or (iii) the tower has toppled.

5) Tower and robot specifications. The simulated tower is composed of the same number and similar distribution of movable versus immobile blocks as the real tower. This is due to slight perturbations to weight distribution resulting from small tolerances in the height of the blocks. The relative dimensions of the tower and the end effector are consistent for both environments.

The robot's nominal execution loop is to select a block at random and to attempt the push primitive. During the push primitive, the robot either chooses push poses and headings or retracts. If the block is extracted beyond three-fourths of its length, the extract/place primitive is invoked. During a run, the nominal execution loop is continued until a termination criterion is met. A key challenge is that movable versus immobile pieces are indistinguishable before contact; consequently, the robot needs to control the hybrid/multimodal interaction for effective extraction without causing damage to the tower. If the damage compounds, then the tower loses integrity and termination criteria are met earlier. Hence, this problem is a challenging example that motivates the need for abstract reasoning with a fusion of tactile and visual information with a rich representation of physics.

Simulation

Figure 2A depicts our Jenga setup in the MuJoCo (*11*) simulation environment. We used the simulation environment to compare the performance of our proposed approach [hierarchical model abstractions (HMAs)] to several standard baselines (Fig. 2B). Specifically, we chose a feed-forward neural network (NN) as a representative nonhierarchical model-based approach, a mixture of regressions (MOR) model as a generic hierarchical model-based approach, and the proximal policy optimization (PPO) (*12*) implementation of RL as a model-free approach. All models have access to the same set of states, actions, and model predictive controller (MPC) (details in Materials and Methods). Figure 2C depicts the schematics of our proposed approach and the MOR models. The MOR model makes use of latent variables l_t, where t denotes time. Our HMA model uses abstractions denoted with c_t. The states and noisy observations are denoted by s_t and z_t, respectively.

Figure 2B shows the number of blocks successfully extracted in sequence as a function of the number of samples used to learn either a model (HMA, MOR, or NN) or a policy (RL). A sample is an incremental slice of the push trajectory. For the model-based approaches, the samples were collected during an exploration phase in which the robot interacted with the tower and collected states and actions. The exploration phase followed the robot's nominal execution loop using the exploration policy. Once data collection was complete, a model was trained and its fidelity was evaluated in a model predictive control framework per number of samples over an unseen set of test towers. For the purpose of evaluation, the test towers were uniform for all models. For reference, a complete push was composed of 45 steps where a sample was measured at each step. A single sample took about 2 s to collect experimentally and 0.3 s in simulation. The robot could interact with a total of 45 blocks for a new tower. We found empirically that about 47% of pieces move in a real-world random tower and emulated this ratio in the simulation environment. Consequently, the robot can extract, on average, 21 blocks for an unperturbed tower of 3 by 18 (the last 3 layers are prohibited by the rules). This value provides a reasonable goal against which performance can be evaluated.

The proposed approach reached the expected number of successful consecutive extractions within 100 samples. The MOR model was next to achieve the maximum score, requiring an order of magnitude more samples. The feed-forward NN saturated in performance, falling short of the expected maximum extractions. Upon closer inspection, we found that this model was unable to reliably predict the multimodal interactions and behaved either too conservatively or too recklessly. The RL algorithm was the slowest in convergence. Per run, all approaches were presented with new towers at random, which explains, in part, why the RL algorithm took a large number of samples to converge: The space of possible tower configurations was very large.

Experiments

In our experimental setup, the robot had access to an Intel RealSense D415 camera (RGB) and an ATI Gamma six-axis force/torque sensor mounted at the wrist (Fig. 1A). These two modalities provided noisy approximations of the current pose of the pieces in the tower and the forces applied to the robot (details in Materials and Methods). We used the robot's forward kinematics to estimate the pose of the gripper. To estimate the pose of the fingertips, we computed the deflection of the fingers with respect to the gripper by using the measured force applied to the finger and known compliance parameters. We

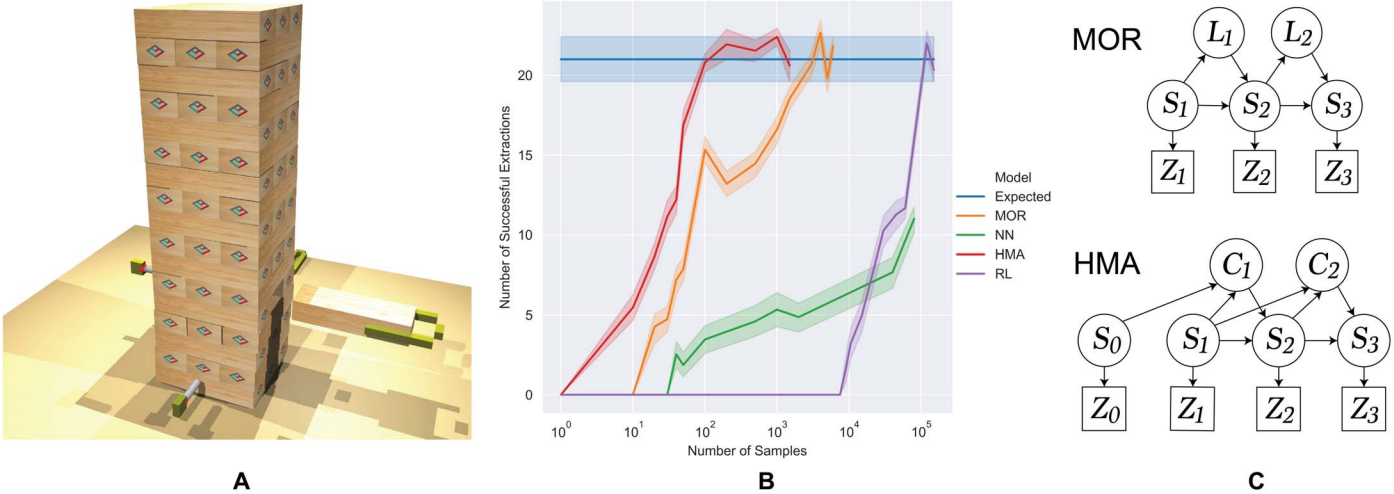

Fig. 2. Jenga setup in simulation and the baseline comparisons. (**A**) The simulation setup is designed to emulate the real-world implementation. (**B**) Learning curve of the different approaches with confidence intervals evaluated over 10 attempts. Solid lines denote the median performances; shadings denote one standard deviation. (**C**) Visual depiction of the structure of the MOR and the proposed approach (HMA).

used the experimental setup to demonstrate the fidelity of the proposed approach.

The failure criteria in the experimental setting were expanded to include tower rotation or displacements exceeding 15° and 10 mm, respectively. This criterion was imposed by the poor predictions made in the vision systems beyond these values. The exploration strategy was also modified to include a hand-coded supervisory algorithm that attempted to mitigate damage to the tower using measurements of poses and forces but was tuned to allow mistakes. Table 1 shows the robot's performance before and after exploration. Here, a successful push is one in which the robot is able to push a block to a desired end goal without dropping it outside the tower or causing excessive damage. A successful extraction is one in which the robot is able to pull the block free after a push without damaging the tower, and a successful placement is placing the block on top of the tower without damage.

The robot showed an appreciable improvement in block extraction from 56 to 88%. The most noticeable gain was in side-block extraction, doubling the success rate from 42.6 to 78.3%. The results suggest that middle-block extraction is considerably easier than the side block extraction because of constrained motion and the favorable weight distribution of the tower. The robot was able to displace 42.7% of the blocks, close to the empirical average of a random tower.

The two main failure modes for the extraction of blocks were as follows: (i) excessive forces applied to the tower (characteristic of failing to identify block behaviors) and (ii) poorly controlled extraction of blocks (characteristic of poor predictive ability). The first failure mode often resulted in either a large tower perturbation such that the vision system was no longer reliable or tower collapse. The second failure mode often led to blocks either dropping outside the tower or ending in configurations that were difficult to grasp or occluded to the camera.

Model learning

In this study, we represented the physics of the tower by using a hierarchical probabilistic model (Fig. 2C) with the structural composition similar to that of dynamic Bayesian networks. We used a top-down bottom-up learning approach to learn abstractions and physics for the tower. This methodology was inspired by models of cognition, in particular "concept learning," and the intuitive abstractions that humans develop to facilitate complex manipulation skills (see Discussion).

In top-down learning, the objective is to build abstractions from the physics of the tower. This approach is an instance of latent variable learning, where the variable identifies the type of macrobehavior. Specifically, in top-down learning, abstractions are acquired before learning detailed motion models and explicitly encode temporal macrobehaviors of blocks. Figure 3 shows a low-dimensional representation of the recovered abstractions through clustering in the relevant features space (details in Materials and Methods). Because the clusters have no labels, we intuited their semantic meanings from inspection of trajectory traces in force and visual domains. The green cluster denotes traces where the robot was not in contact with any block, in particular at the beginning of the trajectory where measured forces are negligible and blocks do not move. The gray cluster denotes blocks that resisted motion and were stuck, exhibiting large resistive forces and little to no translation. The blue cluster denotes blocks that moved fairly easily (large displacements) and exhibited negligible resistance. The yellow cluster denotes blocks that moved but offered meaningful resistance to the robot.

Bottom-up learning refers to learning explicit state-transition models, factored by the abstractions, using sensory data. We used a probabilistic model, here a Bayesian neural network (BNN), to model the conditional distribution of future states given current states and actions in the joint force and visual domains (see Materials and Methods). The BNN was trained on the data collected during exploration.

Figure 4 depicts two examples of the physics learned by these models. Figure 4A depicts the analytical friction cone between the fingertip and block overlaid with the predicted normal and tangential forces given their current measured values. This cone was computed assuming Coulomb friction and rigid point contact between

Table 1. Summary statistics for exploration and learned physics. A comparison of the performances of the robot using the exploration strategy and the learned model.

Block position	Action	Exploration		Learned	
		Attempts	Successes	Attempts	Successes
	Push	403	172 (42.7%)	203	96 (45.8%)
All	Extract	172	97 (56.4%)	93	82 (88.2%)
	Place	97	85 (87.6%)	82	72 (87.8%)
	Push	288	122 (42.4%)	133	69 (51.9%)
Side	Extract	122	52 (42.6%)	69	54 (78.3%)
	Place	52	44 (84.6%)	54	49 (90.7%)
	Push	115	50 (43.5%)	70	33 (47.1%)
Middle	Extract	50	45 (90.0%)	33	28 (84.8%)
	Place	45	41 (91.1%)	28	23 (82.1%)

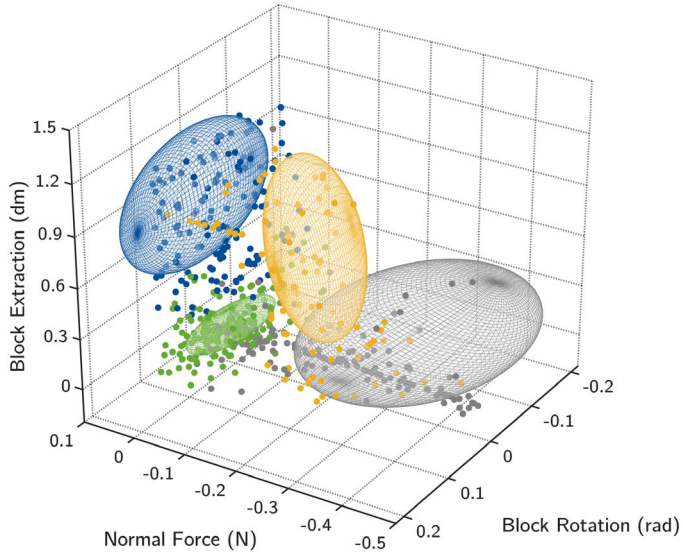

Fig. 3. Concepts learned from exploration data. Means and covariances of the four clusters are projected to the space of "normal force (N)," "block rotation (rad)," and "block extraction/depth (dm)." The four clusters carry intuitive semantic meanings, and we refer to them as follows: green, "no block"; gray, "no move"; blue, "small resistance"; and yellow, "hard move."

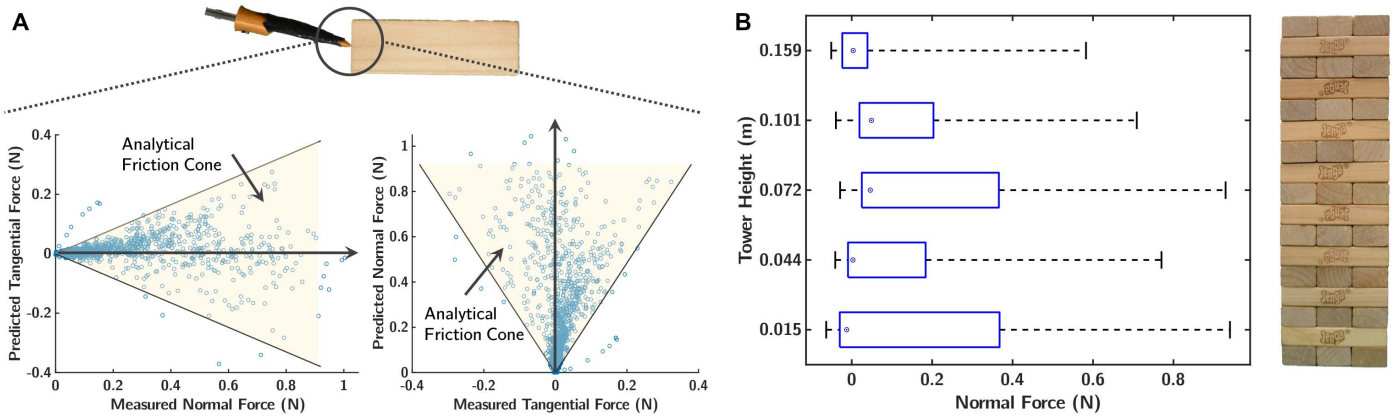

Fig. 4. Learned intuitive physics. (**A**) Overlay of the analytical friction cone and predicted forces given the current measurements. The friction coefficient between the finger material (PLA) and wood is between 0.35 and 0.5; here, we use 0.42 as an approximation. (**B**) Normal force applied to the tower as a function of the height of the tower. Each box plot depicts the minimum, maximum, median, and standard deviation of the force measures.

the fingertip and block. Under these assumptions, any transferable force between the finger and block must lie on the boundary or in the interior of these cones. The predictions of the models are in good agreement with the cone, implying that it has learned some latent representation of friction. We note that the friction cone is invariant to the abstractions, and the predictions of the models reflect this fact well, implying coherency across models and abstractions.

Figure 4B shows the resistive force of blocks as a function of the tower height. The model is able to capture the intuitive tendency of block extraction resistance to decrease with tower height (because of the decrease in effective remaining weight on top of each block). The abstractions facilitate this by factoring the state space between macro-block behaviors and efficiently differentiating between movable versus immobile blocks.

Inference, controls, and planning

The abstractions offer coarse but intuitive and interpretable explanations of physics modes. Hence, knowledge of the particular mode can be effectively leveraged for controls and planning.

As a first step, the robot must infer the abstraction from noisy sensor measurements. Once learned, our representation provides a joint distribution over the states and abstractions, i.e., a generative probabilistic model of physics. We used Markov chain Monte Carlo (MCMC) sampling with Hamiltonian dynamics to approximate the posterior distribution over states and abstractions given noisy measurements. Figure 5 shows two examples of the inferences of block types made by the robot during a push trajectory. The force and displacement plots show the prediction of the state for each abstraction. The abstraction probabilities are directly related to how well they are able to explain the observations.

In the case of a block that does not move (Fig. 5A), since initially there are no changes in force or configuration, the robot assigns the most probability mass to "no block" and some probability to "easy move" as the force measurements match fairly well. As the robot continues to push, the force climbs but configuration does not change, and the probabilities of "no move" and "hard move" increase. The "hard move" abstraction does not predict the continued climb in force with no displacement and loses likelihood in the next few steps.

Figure 5B shows a block that moves with very little resistance. During the initial phase, because of the lack of force and displacement, the robot assigns the most probability to "no block." As the push continues, displacements increase, whereas forces do not, shifting the belief to "easy move" with very low probabilities for the other abstractions.

Qualitatively, the inference is mostly driven by force measurements during the initial phase of a push because of the relatively small changes in configuration. Critically, the perception algorithm has the highest uncertainty in the beginning of the push, because the mask computed for the block is the least informative and many hypotheses explain the current pose. Configurations play a more decisive role further along the push as they become more certain and discriminative between abstractions. This observation highlights the importance of inference and modeling in the joint domain of visual and tactile modalities.

To use the abstractions in a control or decision-making framework, we first need an association between desirable abstractions and actions. To incorporate this information, we add a cost term to the MPC, where actions resulting in increased probability of "no move" are penalized. This approach is very similar to standard reward shaping for model-free RL frameworks. A desired policy is one that leads to minimal damage to a tower. We impose this by assigning a large cost to tower perturbations that is effectively an implicit encoding of our human intuition.

Figure 6 shows an example trajectory for the block and the controller actions. Here, the robot attempts to push the block to a desired end configuration using its MPC. Using the inferred abstraction and states, the robot forward predicts the costs of sequences of actions over a time horizon, then executes the first action from the sequence incurring the smallest cost and repeats. The fidelity of the representation in forward prediction coupled with knowledge of the abstraction provides effective fine-grained control of block motions. This, in turn, significantly mitigates dropping blocks outside the tower or toppling, allowing runs to continue for longer.

DISCUSSION

Here, we take a top-down bottom-up approach to manipulation skill learning. This approach to top-down learning is inspired by "concept learning" as studied in cognitive science. A "concept" is a type of abstraction composed of a typical feature set. We naturally categorize objects and ideas into concepts, for example, doors and happiness.

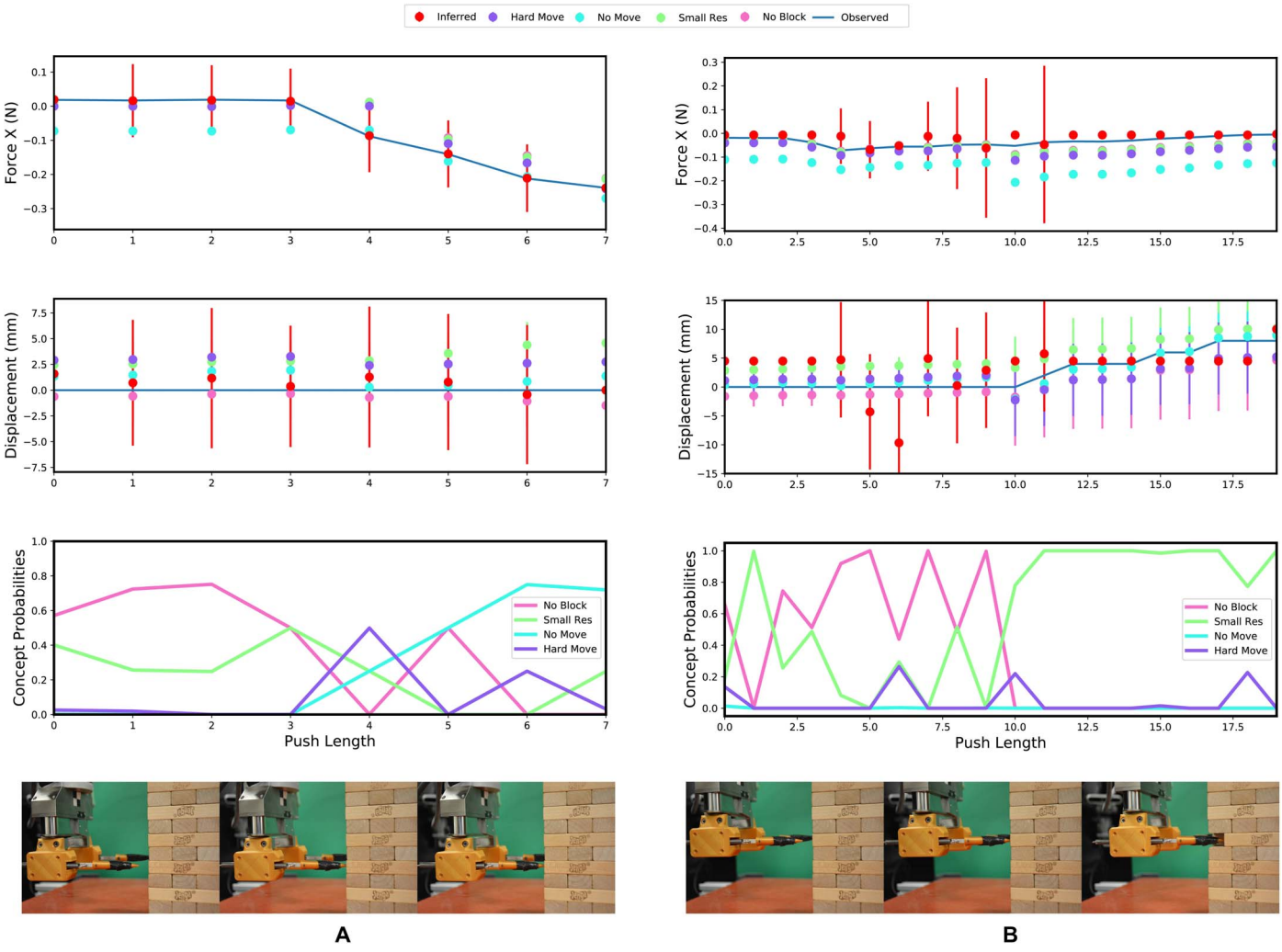

Fig. 5. Inference using the learned representation. Evolution of the beliefs of the robot as it interacts with the tower. (**A**) For a block that is stuck. (**B**) For the block that moves easily. Error bars indicate 1 SD.

Concepts are not measurable; rather, they provide useful abstractions for categorization without the need to specify fine-grained details of instances.

When playing Jenga, we intuitively categorize blocks based on their general behaviors. For example, we may refer to blocks that do not move and exhibit large resistance (the typical feature set) as "stuck." From experience, we learn that this type of block does not help us make progress in the game. We infer connections between concepts and our objectives, but these abstractions are made independent of the task—rather, they are a latent property of the physics. These coarse abstractions organize our reasoning and facilitate decision-making.

From a robotics perspective, there are two important benefits to learning a few general concepts than learning fine-grained physics models. First, by virtue of implicitly factoring the state space and sequentially learning physics using simpler local models, we can significantly increase the sample efficiency of learning (Fig. 2B). Second, concepts may capture interpretable modalities in physics that can be used in controls and planning (Fig. 3). In this study, we take a non–goal-directed perspective, first determining these abstractions then learning models. This approach is related to (*13*), where a goal-directed latent space representation of the task is inferred in the context of learning from demonstration. Certain manipulation tasks, such as meal preparation, may benefit from other abstractions, such as soft or hard. In (*14*), a latent representation of physical attributes of food items was learned that was then used to prepare a salad.

Mixture models are an alternative approach to learning latent states and predictive models. These models are particularly effective in optimizing for predictive ability by simultaneously computing the number of mixtures and model parameters to maximize a likelihood score. A challenge in mixture models is making effective use of the latent variables for model-based controls and planning, in particular when the number of mixtures is large. For example, in the simulation environment, the number of mixtures ranged from 2 to 11 (determined using fivefold cross-validation), whereas the underlying latent structure did not provide additional insight that we found useful to control or decision-making.

An additional technical challenge in mixture model learning is parameter estimation. Expectation maximization is commonly used for estimation but can be sensitive to output variance, initial conditions, and local minima. Sample-based variational inference approaches

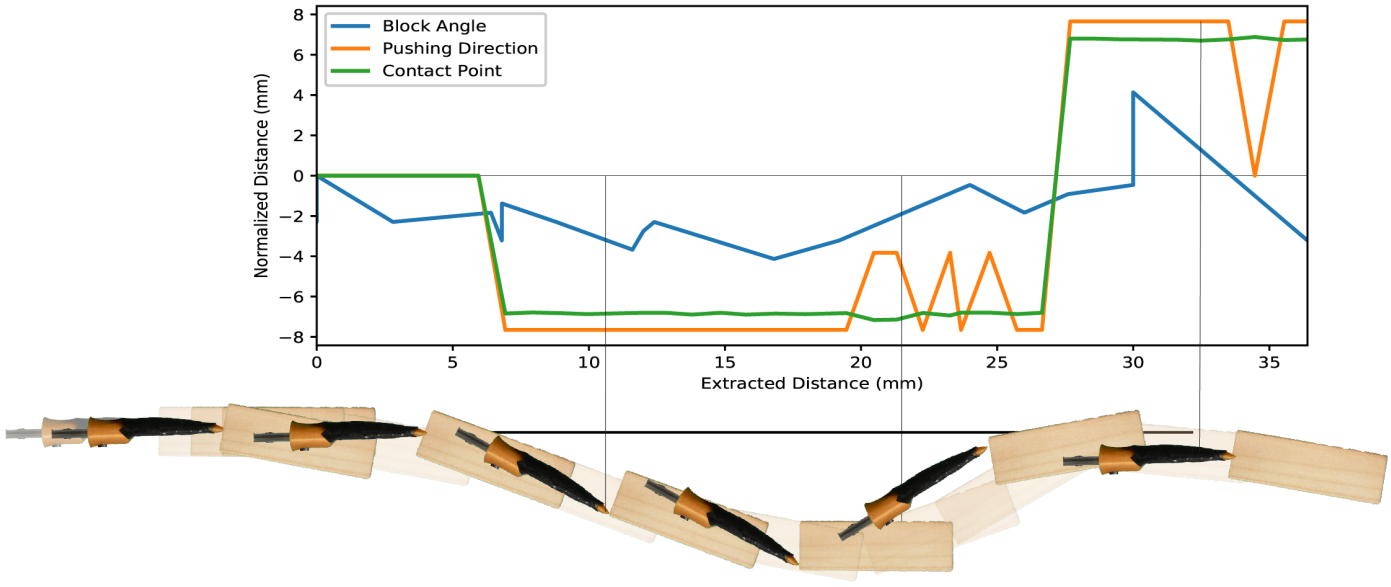

Fig. 6. Controlled block pushing. The robot selects the point on the block and the appropriate angle to push with such that it realigns the block with the goal configuration. Here, the block is beginning to rotate counterclockwise and is starting to move out of the tower. The robot selects a point close to the edge of the block and pushes it back in toward the tower center. We convert angles to normalized distances by scaling with the radius of gyration of the block (0.023 m). We have exaggerated the block translation to illustrate the fine-grained details of motion.

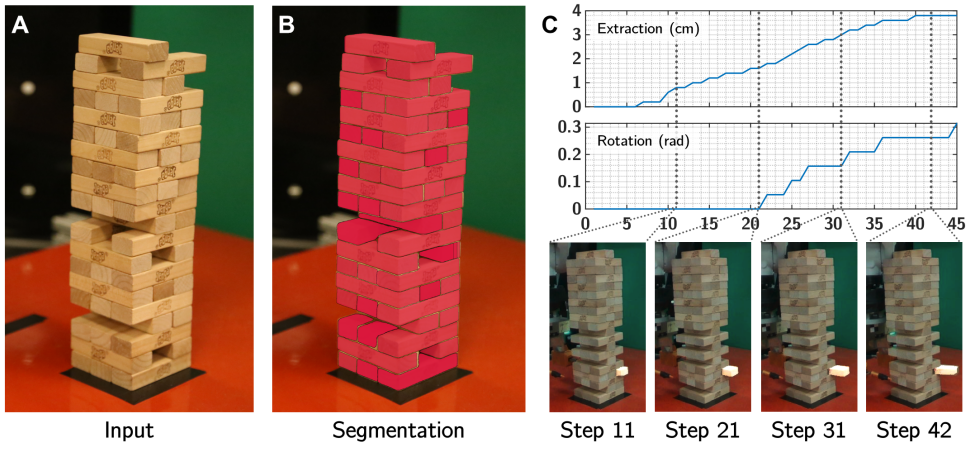

Fig. 7. The vision system. (**A**) We use a 280-by-280 patch with the tower at the center. For clarity, in this figure, we crop the irrelevant regions on both sides. (**B**) Segmented blocks. (**C**) For a single push, we identify the block being pushed and, based on its associated segment, infer the underlying pose and position of the block using an HMM.

do not scale well with data size and model dimensions, in particular because of the simplex constraint on mixture weights and poor mixing of the samples due to the complex topology of the posterior distribution. In both instances, the potentially large number of mixtures further increases the parameter space, requiring more data to learn, and partially explains the results of Fig. 2B.

Mixture model learning is closely related to the idea of dynamical mode learning. Dynamic modes are partitions in state space for which transitions are smooth. Inferring dynamic modes is an important challenge in robotics; however, the level of detail provided by such a description may not be necessary for successful task execution. For example, for a moving block in the Jenga tower, there are at least eight distinct contact formations, four of which are unstable, and each has its own dynamics. The subtle differences in the modes render them difficult to infer, and significant expert knowledge is required to render them useful. Learning coarse abstractions can provide sufficiently descriptive representations of the interactions.

An alternative approach to model-based manipulation skill learning is policy learning. Recent trends in RL for manipulation have mostly focused on learning in the visual domain, leveraging advances in deep RL and computer vision algorithms (*15*, *16*). These algorithms are used to produce autonomously acquired manipulation policies that map from the pixel domain of images to robot actions. Real-world policies learned in the image domain have been shown for planar pushing, bin picking, and insertion (*17*–*21*). These approaches work well for tasks in which the visual data stream provides sufficient information and data collection can be automated effectively.

There are three important challenges in learning policies for manipulation. First, many contact-rich manipulation skills are difficult to automate for large-scale data collection. Jenga is a prime example where tower resets are time intensive; therefore, sample efficiency is an important concern. Second, sim-to-real transfer of policies remains challenging because most simulators use computationally efficient but inaccurate models of frictional interaction (*22*). In the case of Jenga, the block motions rely on very fine microinteractions and pressure distribution variations that are not observable or measurable in practice. Third, tactile information is often intermittent, i.e., making and breaking contact over a short duration. Consequently, effective integration of tactile information, together with the persistent visual stream, is challenging. These challenges are prevalent in many manipulation tasks, such as small parts assembly, warehouse logistics, and disaster response. In such tasks, fine-level reasoning for contact is critical, and mistakes often incur large costs.

The interplay between tactile and visual information is central to manipulation. These two modalities act as complements and provide information in the absence of one another; for example, visual occlusions are a common part of Jenga, where blocks are occluded during manipulation but the tactile modality continues to provide information. Further, tactile feedback can provide high-resolution local information to complement the global but coarse information from vision. In the experimental setting, our perception system is susceptible to errors in pose estimation during the initial phase of the push because the total visible mask of the block is small and several hypotheses may be equally likely. The force stream and proprioception, together with their temporal variation, increase the likelihood of plausible hypotheses for poses despite the occlusion of the end effector due to the tower.

Building coarse abstractions in the joint domain of tactile and visual feedback is central to our approach. These abstractions are partially driven by the temporal change in these signals, motivated by time-varying multimodal mechanics. The effective use of both modalities together with the abstractions leads to significant sample efficiency, allowing the experimental robot to learn tower physics entirely from real-world experience. Furthermore, the abstractions are effectively used for controls and planning, significantly improving the performance of the system. In particular, for Jenga, our controller is able to explicitly use the relation between inferred stuck pieces and tower perturbation to mitigate damage.

MATERIALS AND METHODS

System architecture

In this subsection, we describe the operation of the machine intelligence depicted in Fig. 1B. At a given step, the system records an RGB image using the camera that is processed by the perception algorithm to recover the 6-DOF (degree of freedom) pose of the block. Concurrent with this operation, a low-pass filtered measurement of the force-torque measurement is also recorded. The measured forces and poses are added to a buffer of observations. Next, the inference block uses the observations in the buffer and the learned physics model to estimate the latent states and abstractions.

The estimated states and abstractions are passed to the MPC, where an optimal action (that may involve retraction) is computed using the forward rollouts of the learned physics model. The computed action is then passed to the robot for execution, and the loop is repeated. The game play unit is a hand-coded state machine that sequences the action primitives and monitors the state of the tower using sensor measurements and the inference block. This loop is repeated until a termination criterion is met. In the following subsections, we detail the mechanisms of each block.

Notation

Let $q_t = (x, y, z, \theta_z)_t$ denote the configuration of a block in plane at time t; $f_t = (f_x, f_y)_t$ is the force exerted by the robot (we omitted f_z because it does not change and torques because the interaction between the robot and the block is a point contact and cannot transmit torque); p_t is a measure of the perturbation of the tower defined as net displacement with respect to an unperturbed tower; and $r_t = (J_x, J_y, \phi, w, v)_t$ is the robot pose, push heading, push location on the block in the block frame, and auxiliary action v for retraction. We denote the state as $s_t = (q, f, p, J_x, J_z)_t$ and action as $a_t = (\phi, w, v)_t$.

Baseline models

Mixture of regression

We formulate the MOR model as the linear mixture model:

$$p(s_{t+1}|s_t, a_t) = \sum_{i=1}^{H} \pi_i(\delta_i, s_t, a_t) N\left(s_{t+1}|\alpha_i^T s_t + \beta_i^T a_t, \sigma_i^2\right)$$

Here, the mixture weights are also a function of the input space. We used fivefold cross-validation and expectation maximization to determine the number of mixture components and train the mixture model. We found empirically that to maximize the predictive ability of these models, it is best to train a distinct mixture per output variable.

Feed-forward NN model

This model also uses (s_t, a_t) to predict s_{t+1}. The hidden layer nodes are fully connected layers with ReLu activations. Empirically, we found that a (11, 10, 9, 17) layout for the hidden layers yielded the best predictive performance over the test set for the models that we tried. In this particular model, the input is also directly passed to the final layer. The network was trained in PyTorch with a learning rate of 0.01 and a batch size of 100. Larger batch sizes yielded only marginally better models at the expense of computation time. The resulting models were sensitive to large changes to the learning rate but were consistent in predictive performance for rates close to 0.01.

Reinforcement learning

We used the PPO implementation from the OpenAI Gym (*12*). We passed the state vector s_t to the algorithm and apply the computed a_t to the system. We then evaluated s_{t+1} and the reward function and repeated the cycle. The reward was composed of an accumulating term for positive displacement, a terminal award for a grasped block, an accumulating negative reward for moderate perturbations, and a large negative reward for large perturbations. See the Supplementary Materials for the details of the reward function and hyperparameters.

Concept learning via clustering

We used a Gaussian mixture model with a Dirichlet process (DP) prior over the number of clusters to learn the abstractions. We used the feature set

$$x_t = \{q, \Delta q, f, \Delta f, J, a\}_t$$

where Δ denotes temporal difference and the variables maintain the definitions from the previous subsection. To generate a dataset of

features, we randomly sampled 80 samples from random push actions conducted during the exploration phase. We found empirically that very similar clusters are recovered with as few as 40 samples, although occasionally fewer clusters were found because of random selection of samples. We set the α prior for the DP to 0.7 and let the variational inference algorithm iterate for a maximum of 30 times. Values lower than 0.7 consistently yielded fewer clusters, in particular, the no move, move, and no blocks occur often. On some occasions, the no block and move melded into one. Values higher than 0.7 yielded a larger number of clusters, approaching numbers determined by the mixture models. To check cluster consistency, we repeated the dataset construction and variational inference steps 10 times.

Once the variational inference algorithm concluded, we evaluated the number of clusters by checking the posterior distribution over the mixture weights. We found that the posterior weights for four of the clusters were meaningful and used the mean and covariances of these four clusters as the abstractions in the model.

BNN motion model

We used a BNN to model the state transitions (configurations and forces) in the physics model of the Jenga tower. These models used the same input features as the baseline models with an additional one-hot encoding of the latent variable denoted with c_t, predicted the same set of variables, and computed a probability distribution. We used a probabilistic model to build a generative model of physics to be used with probabilistic inference.

For the network architecture, we used an arrangement of (10, 7, 5) for the hidden nodes. This form of network was chosen to emulate an encoder network, because the various inputs to the network are correlated and potentially span a submanifold of the space. For details, see the Supplementary Materials.

Probabilistic inference and MPC control

We used MCMC to perform probabilistic inference over the learned representation of physics. To sample from the Bayesian hierarchical model, we used a prior over the latent variables using a DP. Because the variables are one-hot encoded in the motion models, we chose our prior hyperparameters such that the DP, when sampled, returned a value close to a class one-hot encoding. For the observations, we assumed a zero mean normal distribution with a variance equal to the variance measured from the sensors. We used PyMC3 to encode the model and ran MCMC given measurements accumulated by the robot at run time. We ran four parallel MCMC chains on four central processing units at run time, each with 3000 samples and a burn-in of 500. One call to the inference algorithm took about 0.5 s. We found that the benefit of running inference for longer periods to yield better estimates of the latent variables was not worth the incurred computational expense.

To control the block motions, we used a sampling-based, greedy MPC. In particular, the MPC assigns a quadratic cost to the distance from goal, change in action, and perturbation of the tower. The controller looks ahead five steps using a learned model and selects the action sequence with the lowest cost. It executes this action and repeats. For details of implementation, see the Supplementary Materials.

Perception

Our visual perception system (Fig. 7) maps a single-color image of the Jenga tower to a collection of blocks, each with its own position and rotation. The system operates in two steps: It first obtains the segment of each block using deep convolutional networks; it then applies template matching to recover the 6-DOF pose of the block. Leveraging state-of-the-art techniques in visual perception and traditional model-fitting algorithms, the system is fast, accurate, and robust to noise.

The vision module first segments out each block from the image. To do this, we used a state-of-the-art convolutional network for this purpose, Mask R-CNN (*23*), with a backbone structure of ResNet-50 (*24*). The camera output has a resolution of 640 by 480 pixels, and we only use a 280-by-280 patch with the tower at the center. During inference, we set the nonmaximum suppression threshold to 0.4 to avoid overlapping segments. We follow the original Mask R-CNN paper for other implementation details (*23*). The segmenting network is learned using a combination of 3000 synthetic images of the Jenga tower rendered in Blender and 150 images of the Jenga tower in the experimental setup. See the Supplementary Materials for further details.

After obtaining the segments, the visual module maps each segment to a three-dimensional block with its position and pose via a hidden Markov model (HMM). The main idea is to find the sequence of block positions and poses that explain visual observations well while maintaining temporal smoothness. Specifically, given segments $M = \{m_i\}$ as our observations, we want to find the underlying states $S = \{s_i\}$ that maximize $\Pi_i P(m_i|s_i)P(s_{i+1}|s_i)$. A state s is parametrized as a quintuple (p, q, x, y, θ), where p represents the block's level of height, q represents its order (left, middle, right) among the blocks at the same level, x and y represent its horizontal and vertical translation with a granularity of 0.002 m, and θ represents its in-plane rotation with a granularity of 3°.

We used the classic Viterbi algorithm for inference. We compute the probability of observation m_i given state s_i using three criteria: the intersection over union between the segment provided by Mask R-CNN and the segment of the block from the template (IOU_s), the intersection between the bounding box of the Mask R-CNN segment and the template segment (IOU_b), and the chamfer distances for the two sets of pixels within the two segments. For details of the transition probability and related cost function, see the Supplementary Materials.

SUPPLEMENTARY MATERIALS

robotics.sciencemag.org/cgi/content/full/4/26/eaav3123/DC1
Additional Materials and Methods
Movie S1. Summary.

REFERENCES AND NOTES

1. E. S. Spelke, K. D. Kinzler, Core knowledge. *Dev. Sci.* **10**, 89–96 (2007).
2. M. T. Mason, Towards robotic manipulation. *Annu. Rev. Control Robot. Auton. Syst.* **1**, 1–28 (2018).
3. R. S. Johansson, J. R. Flanagan, Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nat. Rev. Neurosci.* **10**, 2345 (2009).
4. A. Petrovskaya, O. Khatib, S. Thrun, A. Y. Ng, Touch based perception for object manipulation, in *Proceedings of Robotics Science and Systems (RSS), Robot Manipulation Workshop* (2007), pp. 2–7.
5. M. O. Ernst, M. S. Banks, Humans integrate visual and haptic information in a statistically optimal fashion. *Nature* **415**, 429–433 (2002).
6. K. P. Körding, D. M. Wolpert, Bayesian integration in sensorimotor learning. *Nature* **427**, 244–247 (2004).
7. G. Erdogan, I. Yildrim, R. A. Jacobs, Transfer of object shape knowledge across visual and haptic modalities. *Proc. Annu. Meet. Cognit. Sci. Soc.* **36** (2014).
8. P. W. Battaglia, J. B. Hamrick, J. B. Tenenbaum, Simulation as an engine of physical scene understanding. *Proc. Natl. Acad. Sci. U.S.A.* **110**, 18327–18332 (2013).
9. J. Fischer, J. G. Mikhael, J. B. Tenenbaum, N. Kanwisher, Functional neuroanatomy of intuitive physical inference. *Proc. Natl. Acad. Sci. U.S.A.* **113**, E5072–E5081 (2016).

10. T. D. Ullman, E. Spelke, P. Battaglia, J. B. Tenenbaum, Mind games: Game engines as an architecture for intuitive physics. *Trends Cogn. Sci.* **21**, 649–665 (2017).
11. E. Todorov, T. Erez, Y. Tassa, MuJoCo: A physics engine for model-based control, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2012), pp. 5026–5033.
12. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms. arXiv:1707.06347 [cs.LG] (20 July 2017).
13. S. Calinon, F. Guenter, A. Billard, On learning, representing, and generalizing a task in a humanoid robot. *IEEE Trans. Syst. Man Cybern. B Cybern.* **37**, 286–298 (2007).
14. M. C. Gemici, A. Saxena, Learning haptic representation for manipulating deformable food objects, in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2014), pp. 638–645.
15. K. Arulkumaran, M. P. Deisenroth, M. Brundage, A. A. Bharath, Deep reinforcement learning: A brief survey. *IEEE Signal Process. Mag.* **34**, 26–38 (2017).
16. P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, S. Levine, Learning to poke by poking: Experiential learning of intuitive physics, in *Proceedings of Advances in Neural Information Processing Systems* (NIPS, 2016), pp. 5074–5082.
17. Y. Gao, L. A. Hendricks, K. J. Kuchenbecker, T. Darrell, Deep learning for tactile understanding from visual and haptic data, in *2016 IEEE International Conference on Robotics and Automation* (ICRA, 2016), pp. 536–543.
18. A. Kloss, S. Schaal, J. Bohg, Combining learned and analytical models for predicting action effects. arXiv:1710.04102 [cs.RO] (11 October 2017).
19. S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, D. Quillen, Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Rob. Res.* **37**, 421–436 (2018).
20. J. Mahler, K. Goldberg, Learning deep policies for robot bin picking by simulating robust grasping sequences, in *Proceedings of the 1st Conference on Robotic Learning* (CoRL, 2017), pp. 515–524.
21. S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **17**, 1334–1373 (2016).
22. N. Fazeli, S. Zapolsky, E. Drumwright, A. Rodriguez, Fundamental limitations in performance and interpretability of common planar rigid-body contact models. *CoRR* **abs/1710.0** (2017).
23. K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, in *Proceedings of the IEEE International Conference on Computer Vision* (IEEE, 2017), pp. 2980–2988.
24. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 770–778.

Acknowledgments: We thank L. P. Kaelbling and T. Lozano-Perez for the insightful discussions related to the project. **Funding:** The research was supported by NSF grants 1231216 and 1637753. **Author contributions:** N.F. contributed to the main ideas of the paper, experimental and simulation implementations, writing, and hierarchical learning. M.O. contributed the lion's share of the coding and design in experimental and simulation setup. J.W. and Z.W. contributed the perception algorithm and insights into model learning. J.B.T. contributed to writing and provided deep insights and guidance on hierarchical learning and implementation. A.R. contributed to writing and provided deep insights into manipulation, system architecture, and physics models. **Competing interests:** The authors state that they have no competing interests. **Data and materials availability:** The code base and data needed to evaluate the conclusions of this paper are available at https://github.com/mcubelab/mo_jenga_public.

Submitted 5 September 2018
Accepted 4 January 2019
Published 30 January 2019
10.1126/scirobotics.aav3123

Citation: N. Fazeli, M. Oller, J. Wu, Z. Wu, J. B. Tenenbaum, A. Rodriguez, See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion. *Sci. Robot.* **4**, eaav3123 (2019).

See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion

N. Fazeli, M. Oller, J. Wu, Z. Wu, J. B. Tenenbaum and A. Rodriguez

Sci. Robotics **4**, eaav3123.
DOI: 10.1126/scirobotics.aav3123