Curiosity-Driven Learning of Abstract Plan Feasibility

Michael Noseworthy*, Caris Moses*, Isaiah Brand*,
Sebastian Castro, Leslie Kaelbling, Tomás Lozano-Pérez, Nicholas Roy
MIT, CSAIL

***Abstract*—Long horizon sequential manipulation tasks are effectively addressed hierarchically: at a high level of abstraction a planner searches over abstract action sequences, and when a plan is found, lower level motion plans are generated. This approach hinges on a reliable prediction of *Abstract Plan Feasibility* (APF), which is challenging because the outcome of a plan depends on real-world phenomena that are difficult to model, such as noise in estimation and execution. We apply an active learning strategy that leverages an *infeasible subsequence property* to prune candidate plans, allowing our system to learn from less data. We evaluate our method in simulation and on a real Franka Emika Panda robot in a stacking domain where objects have non-uniform mass distributions. The robot learns an APF model in four hundred self-supervised interactions, and uses the learned model effectively in multiple downstream tasks. An accompanying video can be seen at https://bit.ly/3bsTYn3.**

I. Introduction

Long horizon sequential manipulation tasks still pose a challenging problem for robotic systems. Finding a plan to achieve a task in these domains consists of reasoning over large spaces that include discrete action plans, as well as low level continuous motion plans. These problems can be effectively addressed hierarchically: at the highest level of abstraction the system searches over plausible *abstract* action sequences, and at the lower level it plans for a detailed *concrete* motion plan and object interactions. Further computational efficiencies can be gained if we lazily [13] postpone concrete planning until we have a complete abstract plan that is likely to succeed, avoiding the need to query the concrete planner multiple times.

The success of this lazy strategy hinges on our ability to predict whether an abstract action sequence will be feasible to execute. In this work, we call an abstract action sequence feasible if both the concrete planner returns a solution and this solution is reliably executed in the real world. Fortunately, even an approximately correct estimator of *abstract plan feasibility* (APF) can offer huge computational advantages during planning. In some cases it may be possible to approximate APF via coarse-grained simulation. However, this strategy still requires a dynamics model, which may not capture phenomena needed to accurately predict feasibility in the real world.

Instead, we explore a model learning strategy to predict APF by exploring the space of real plan executions without a specific planning problem or task at hand — a form of curious exploration [21]. However, data efficiency is a primary concern

*Equal contribution.

The authors would like to thank Rachel Holladay for generously contributing her time to help us get started using the Panda robot. This research was sponsored by the Honda Research Institute.

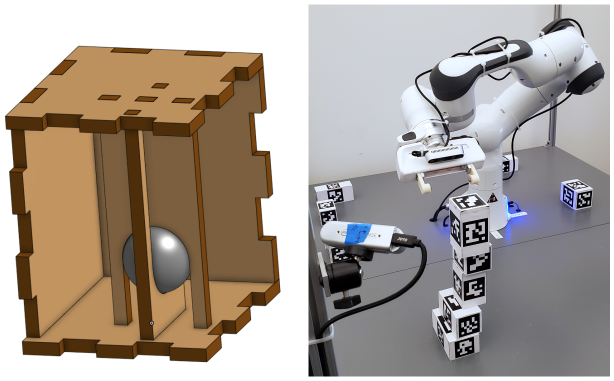


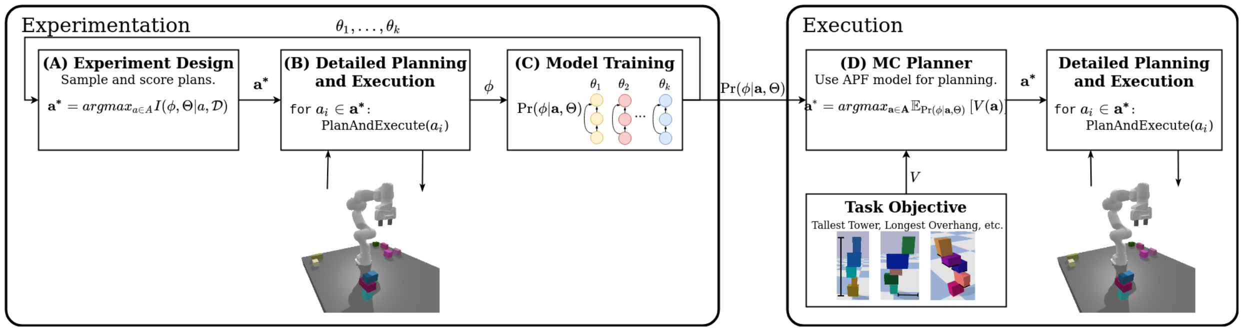
Fig. 1: Left: Task block with an internal weight that alters its mass distribution. Right: The Franka Emika Panda robot constructing a tower. Blocks are detected using robot mounted and external cameras.

in enabling real robot learning of feasibility models, so we note that some observations may be more valuable than others. To determine how informative a plan is with respect to the learned APF model, we adopt an information-theoretic active learning approach [20, 15]. To generate candidate plans, we exploit an important property of abstract action sequences: for an action sequence $(a_1, \ldots, a_n)$, if any prefix $(a_1, \ldots, a_i)$ is infeasible, then any longer prefix $(a_1, \ldots, a_j)$ for $i < j \leq n$ is also infeasible. This *infeasible subsequence property* gives us leverage during data acquisition. A complex plan instance may contain many elements that are highly informative for model learning, but will never be experienced because early elements in the plan will fail with high probability.

We apply this active learning strategy to the concrete problem of stacking blocks with a real robot, where the blocks are each unique, and have non-uniform mass distributions. The robot autonomously designs, plans, and executes experiments to learn a feasibility model using a Franka Emika Panda robot arm (Figure 1). The robot is also capable of resetting the world state after each experiment, enabling continuous autonomous experimentation. The learned feasibility predictor is later used to build towers with previously unseen blocks that satisfy several different objective functions, including the tallest possible tower or the tower with the longest overhang.

In summary, our contributions are:

- A method to learn an *Abstract Plan Feasibility* model by synthesizing hypothetical plans;
- A data acquisition approach which leverages the *infeasible subsequence property* when sampling potential plans;
- A robotic system which conducts autonomous self-supervised learning via integrated perception, experimen-



Experimentation
$\theta_1, \ldots, \theta_k$
(A) Experiment Design
Sample and score plans.
$\mathbf{a}^* = argmax_{a \in A} I(\phi, \Theta | a, \mathcal{D})$
$\mathbf{a}^*$
(B) Detailed Planning and Execution
for $a_i \in \mathbf{a}^*$: PlanAndExecute(a_i)
ϕ
(C) Model Training
$\Pr(\phi | \mathbf{a}, \Theta)$
θ_1 θ_2 θ_k
$\Pr(\phi | \mathbf{a}, \Theta)$
Execution
(D) MC Planner
Use APF model for planning.
$\mathbf{a}^* = argmax_{\mathbf{a} \in \mathbf{A}} \mathbb{E}_{\Pr(\phi|\mathbf{a},\Theta)} [V(\mathbf{a})]$
$\mathbf{a}^*$
Detailed Planning and Execution
for $a_i \in \mathbf{a}^*$: PlanAndExecute(a_i)
V
Task Objective
Tallest Tower, Longest Overhang, etc.

Fig. 2: The proposed system for learning *Abstract Plan Feasibility* (APF) operates in two phases. **Experimentation Phase** (left) The robot iteratively designs and executes experiments that improve its APF model. **(A)** Using its current model, the robot selects the abstract action sequence, $\mathbf{a}^*$, that maximizes information gain over the APF model. **(B)** The robot then computes and executes a concrete motion plan for $\mathbf{a}^*$. **(C)** After observing the true plan feasibility, ϕ, the robot uses this new labeled data to update its APF model, represented as an ensemble of neural networks. **Execution Phase** (right) Once an APF model has been learned, the robot can use it to perform various tasks, such as build the tower with the longest overhang.

tation, planning, and execution.

II. Problem Formulation

Our objective is to learn a model that predicts the success of an abstract action sequence when it is executed by the robot. That is, to learn the parameters $\Theta \in \mathbb{R}^d$ that predict

$$\Pr(\phi \mid \mathbf{a}; \Theta),$$

where $\phi \in \{0, 1\}$ is the success of the sequence of abstract actions, $\mathbf{a} = (a_1, \ldots, a_n)$. Furthermore, we wish to learn Θ using as few labeled action sequences $(\mathbf{a}, \phi)$ as possible.

To learn this APF model, our system operates in two phases as illustrated in Figure 2. In the *experimentation phase*, the robot curiously explores the space of possible plans, learning the parameters of the APF model; in the *execution phase*, the robot is given specific goals to achieve, and uses the learned APF model to efficiently plan over abstract action sequences.

III. Active Learning of Abstract Plan Feasibility

Collecting data on real robot platforms is both time and cost-intensive. In order to minimize the amount of data needed to learn the APF model, we take an information theoretic approach to active learning [15]. Efficient active learning requires: (1) a model class that captures uncertainty over model parameters, (2) a way to score unlabeled plans based on how informative they may be, and (3) a method of generating potentially informative plans. We discuss each of these in turn.

Abstract Plan Feasibility Model Our APF model, $\Pr(\phi \mid \mathbf{a}; \Theta)$, aims to capture the uncertainty in the underlying stochastic process of predicting the feasibility of abstract action sequences. This uncertainty can be attributed to phenomena such as the robot's motor capabilities, errors in perception, or unmodeled behaviors of the planning process, and is referred to as *aleatoric uncertainty*. Our goal is to learn parameters Θ such that this uncertainty is adequately captured and our model can be leveraged, along with a low level planner, to achieve a goal.

We take a Bayesian approach to learning the model parameters, and maintain a distribution over the parameter space, $\Pr(\Theta)$. This distribution aims to capture the uncertainty we have regarding the accuracy of our predictions, referred to as *epistemic uncertainty*. In general, for complex model classes such as neural network classifiers, an explicit representation of $\Pr(\Theta \mid \mathcal{D})$ for training data $\mathcal{D}$ is difficult to construct or update with new data. We therefore follow the strategy of Beluch et al. [1] and represent this uncertainty with an ensemble of N models, $(\theta_1, \ldots, \theta_N)$, where $\theta_i \in \mathbb{R}^d$. Initial parameters are drawn independently at random and are updated to incorporate new data via gradient descent.

We use *graph neural networks* (GNNs), which exploit parameter sharing to model global properties of plans of arbitrary size using a fixed-dimensional parameterization, Θ.

Entropy Reduction Following [20, 3], we guide our active learning by picking a sequence of data $\mathcal{D}$ that maximally reduces the entropy of $\Pr(\Theta \mid \mathcal{D})$. The general problem of designing a sequence of experiments to minimize entropy is a difficult sequential decision-making problem. Fortunately, due to sub-modularity, a myopic approach that considers only the next experiment to conduct can be shown to be a good approximation [6].

Estimating the entropy over a high-dimensional parameter space is expensive, so we follow the approach of Houlsby et al. [15] to reformulate the objective as:

$$\mathbf{a}^* = \underset{\mathbf{a} \in \mathbf{A}}{\operatorname{argmax}} \ \mathrm{I}(\Phi : \Theta \mid \mathcal{D}, \mathbf{a}) \tag{1}$$

$$= \underset{\mathbf{a} \in \mathbf{A}}{\operatorname{argmax}} \ \mathrm{H}(\Phi \mid \mathcal{D}, \mathbf{a}) - \mathbb{E}_{\Theta \sim \Pr(\cdot \mid \mathcal{D})} \left[\mathrm{H}(\Phi \mid \mathbf{a}; \Theta) \right], \tag{2}$$

allowing the computation of entropies to take place in the lower-dimensional label space, Φ. This is known as *Bayesian Active Learning by Disagreement*, or BALD.

The BALD objective invites an appealing interpretation: maximizing the first term encourages selecting an $\mathbf{a}$ that our model is overall uncertain about, and minimizing the second term encourages selecting an $\mathbf{a}$ for which the individual models in $(\theta_1, \ldots, \theta_N)$ can make confident predictions about the outcome ϕ. If we think of the overall uncertainty as a combination of *epistemic* and *aleatoric uncertainty*, then this objective seeks an experiment with high overall uncertainty and low *aleatoric uncertainty*, which therefore has high *epistemic uncertainty*.

Using an ensemble of equally weighted parameter vectors $(\theta_1, \ldots, \theta_N)$ to represent $\Pr(\Theta \mid \mathcal{D})$ allows us to compute a global feasibility prediction, $\widehat{\Pr}(\Phi = \phi \mid \mathbf{a}; \Theta)$, as well as find the experiment that optimizes the estimated BALD objective in the form:

$$\mathbf{a}^* = \underset{\mathbf{a} \in \mathbf{A}}{\operatorname{argmax}}\ \mathrm{H}(\widehat{\Pr}(\Phi \mid \mathbf{a}; \Theta)) - \frac{1}{N} \sum_{i=1}^{N} \mathrm{H}(\Pr(\Phi \mid \mathbf{a}; \theta_i)). \tag{3}$$

Sampling Strategies Now that we have established an informational score for experiments, we consider several sampling strategies for optimizing over $\mathbf{A}$, the set of plans up to a fixed length L. Maximizing the BALD objective over the entire set $\mathbf{A}$ is difficult because we need to consider all discrete plans up to length L, as well as all possible assignments to each continuous abstract action parameter.

Incremental: We consider a strategy where we only consider plans $(a_1, \ldots, a_n)$ for which we have already observed the prefix to be feasible. In other words, the prefix $(a_1, \ldots, a_{n-1})$ together with result $\phi_{1:n-1} = \mathbf{1}$ are in the current data set $\mathcal{D}$.

Greedy: Another strategy requiring fewer samples is a *greedy* approach, in which we select the next action a_n which maximizes the BALD objective, given that we have already optimistically constructed $a_{1:n-1}$. This strategy does not take into consideration that if a plan fails early, we do not get to learn from the full plan execution.

Random: We consider a strategy where we consider randomly generated sequences of actions of varying length.

IV. Implementation

Domain We have implemented this framework in a block stacking domain (see Figure 1). The world consists of a 7-DOF Franka Emika Panda robot and a set of cuboids with non-uniform mass distributions with which it can interact.

To detect blocks, we use three RealSense D435 depth cameras—two mounted on static frames and the third on the robot wrist for pre-grasp pose refinement. To unambiguously indicate identity and orientation, each block is patterned with a unique ArUco marker on each face.

We use PDDLStream [10], which integrates PDDL task-level planning with lower-level motion planning. The planning domain describes actions including picking and placing objects and moving the arm through free space, while a Bidirectional-RRT [17] in joint configuration space performs collision-aware motion planning with a surrogate world model in PyBullet [4].

If the state of the world prevents the robot from successfully planning or executing a plan, for example a block falling off the table, human intervention may be required. Once such issues are manually resolved the robot can resume planning.

Learning As discussed in Section III, the distribution over APF model parameters is represented by an ensemble of GNNs. The input to the GNN is an abstract action sequence, $\mathbf{a}$. Each action a_i consists of an encoding of the target block's parameters and placement pose. In our experiments, 10 networks are used in the ensemble. Each individual network is randomly initialized and trained using the binary cross entropy loss function with early stopping according to the loss on a validation set, which is also collected actively.

Before active experimentation, each model in the ensemble is initialized by training on the same dataset (shuffled differently for each) of 40 randomly generated towers. At each iteration of the experimentation phase, the top 10 most informative towers are chosen and labeled by attempting to build each with the robot. 20% of the collected data is added to a validation set and the remainder to the training set.

When evaluating task performance, we randomly select 5 blocks from a set of novel blocks and execute the best tower found by the Monte-Carlo planner (that uses all 5 blocks), given the task objective and our *Learned* APF model. The reward received from executing this tower is used to calculate normalized regret, which is the difference between this received reward and the largest reward of a stable tower considered by the planner (found using an analytical model). If a tower is unstable, we assign a reward of zero.

We evaluate our method on the following tasks:

1) *Tallest Tower:* Construct the tallest possible tower.
2) *Maximum Unsupported Area:* Construct the tower where each block has as much area possible unsupported by the block below it.
3) *Longest Overhang:* Construct the tower with the maximum distance from the center of geometry of the bottom block to the furthest vertical side of the top block.

V. Evaluation

Impact of Sampling Strategy Figure 3 shows the task performance of models trained using the three sampling strategies described in Section III. Each strategy has results aggregated from 3 independent training runs and 50 task evaluations per run. Different sets of blocks are used between training and evaluation.

The *incremental* method performs well, successfully minimizing regret with minimal variance across runs. The naive *greedy* strategy performs poorly, and is only able to achieve decent performance on the *Tallest Tower* task after seeing roughly 800 training towers, likely due to the fact that it is not considering the feasibility of subtowers when searching the action space, just greedy single-block placements. This highlights the importance of considering the *infeasible subsequence property* when sampling and scoring plans. The *random* method is not leveraging active learning, and at each iteration just trains on randomly generated towers. As a result it has high variance on all tasks.

The *Maximum Unsupported Area* and *Longest Overhang* tasks are more challenging for the agent because they require deep understanding of the tower stability decision boundary, while the *Tallest Tower* task only requires a rough understanding of how to build stable towers with high confidence. These results show that in spite of the difficulty of the first two tasks, the active learner is able to improve its understanding of the

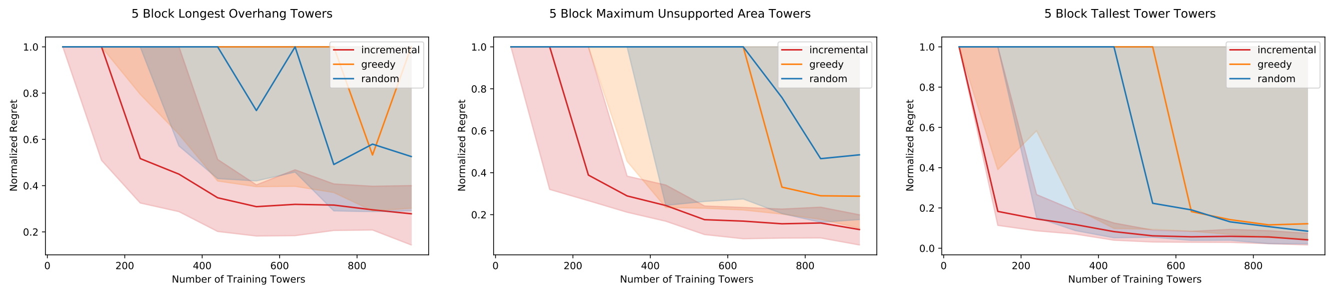


Fig. 3: A comparison of sampling strategies on different tasks. Each evaluation consists of 3 separate APF model-learning runs, and each point is the Median Normalized Regret of 50 individual planning runs per learned model. The shaded regions show 25% and 75% quantiles.

	Tallest Tower			Longest Overhang			Max Unsupported Area		
APF Model	**Regret**	**Stable Regret**	**# Stable**	**Regret**	**Stable Regret**	**# Stable**	**Regret**	**Stable Regret**	**# Stable**
Analytical	0.50	0.00	5/10	0.70	0.00	3/10	0.90	0.00	1/10
Simulation (5mm noise)	0.31	0.01	7/10	0.48	0.35	8/10	0.33	0.16	8/10
Learned	0.14	0.05	9/10	0.33	0.33	10/10	0.24	0.24	10/10

TABLE I: Real robot task performance when using different *Abstract Plan Feasibility* models. The *Learned* model was trained with data collected through active learning on the real Panda robot. The *Analytical* and *Simulation* models calculate feasibility using the known underlying dynamics but use no noise and simple noise models respectively.

decision boundary well enough to perform tasks with very low regret and low variance.

Real Robot Experiments We give results for executing our active learning pipeline on a real Panda robot. In total, the robot built 400 towers while training over a period of 55 hours. We generated candidate experiments using the *incremental* strategy and produced stability labels by observing the outcome with the cameras. During the evaluation phase, we use a separate set of 10 evaluation blocks. We compare the learned APF model to two baselines: (1) a hand-engineered model of plan feasibility that calculates whether a candidate tower is stable in a noiseless world, and (2) a noisy simulator model, which predicts a tower is feasible only if the candidate tower is also stable to 10 normally distributed perturbations (standard deviation of 5mm for each block placement).

Quantitative results are given in Table I and qualitative results are shown in Figure 4. From these results, it can be seen that the *Analytical* model can build towers with high reward when the tower is stable, but the towers it chooses to build are rarely stable across all three tasks. However, the *Simulation* and *Learned* APF models can still build towers with large overhang while considering the effects of noisy action execution on a real robot. While our *Learned* model exhibits slightly higher stable regret, it outperforms the *Simulation* model in stability of constructed towers, and therefore results in overall lower regret when taking stability into account. This is due to the fact that the *Simulation* model makes assumptions about the type of noise distribution (Gaussian only in the plane normal to the table) and its parameters (variance). The *Learned* model, one other hand, learns a more complex noise model that more accurately reflects the real world.

VI. Related Work

Hierarchical Planning A common approach to long-horizon planning problems is to decompose the solution into high-level reasoning over *abstract actions* and lower-level reasoning over *concrete actions* [19, 24, 26]. Recent works predict feasibility of an action as a way to reduce the number of calls to expensive solvers [5, 27].

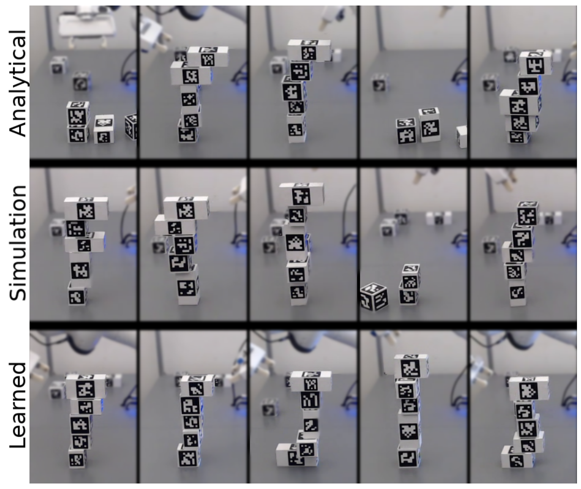


Fig. 4: Towers built for the *Longest Overhang* task when using different *Abstract Plan Feasibility* models. Observe that more towers built using the Analytical and Simulation models were unstable.

Active Learning Active learning [20] is a paradigm that aims to minimize the number of samples needed to learn the target concept. Recently, Gal et al. [9] have extended BALD [15] to complex model domains of deep neural networks using MC-dropout [8]. However, in this work, we follow the approach of Beluch et al. [1] and use an ensemble of deep networks for active learning. These ideas have also been used in model-based reinforcement learning [22, 25].

Stacking Domain *Blocks Worlds* have a long history in artificial intelligence [2, 28, 7]. Computer vision researchers have developed scene understanding algorithms that take into account known geometries and stability properties of objects within the scene [12, 16, 23]. Recent work has shown the ability to predict tower stability from RGB images using deep learning techniques [11, 18, 14].

References

[1] W. H. Beluch, T. Genewein, A. Nurnberger, and J. M. Kohler. The Power of Ensembles for Active Learning in Image Classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] M. Blum, A. Griffith, and B. Neumann. A stability test for configurations of blocks. 1970.

[3] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research (JAIR)*, 4:129–145, 1996.

[4] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2019.

[5] D. Driess, O. Oguz, J. S. Ha, and M. Toussaint. Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[6] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. *An analysis of approximations for maximizing submodular set functions—II*, pages 73–87. Springer Berlin Heidelberg, 1978.

[7] F. Furrer, M. Wermelinger, H. Yoshida, F. Gramazio, M. Kohler, R. Siegwart, and M. Hutter. Autonomous robotic stone stacking with online next best object target pose planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[8] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*. PMLR, 2016.

[9] Y. Gal, R. Islam, and Z. Ghahramani. Deep Bayesian Active Learning with Image Data. In *International Conference of Machine Learning (ICML)*, 2017.

[10] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling. PDDLStream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 2020.

[11] O. Groth, F. B. Fuchs, I. Posner, and A. Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *European Conference on Computer Vision (ECCV)*, 2018.

[12] A. Gupta, A. A. Efros, and M. Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *European Conference on Computer Vision (ECCV)*, 2010.

[13] N Haghtalab, S Mackenzie, AD Procaccia, O Salzman, and S Srinivasa. The provable virtue of laziness in motion planning. *International Conference on Automated Planning and Scheduling (ICAPS)*, 2018.

[14] J. B. Hamrick, K. R. Allen, V. Bapst, T. Zhu, K. R. McKee, J. B. Tenenbaum, and P. W. Battaglia. Relational inductive bias for physical construction in humans and machines. In *the Annual Meeting of the Cognitive Science Society (CogSci)*, 2018.

[15] N. Houlsby, F. Huszar, Z. Ghahramani, and M. Lengyel. Bayesian Active Learning for Classification and Preference Learning. In *NeurIPS Workshop on Bayesian optimization, experimental design and bandits: Theory and applications*, 2011.

[16] Z. Jia, A. C. Gallagher, A. Saxena, and T. Chen. 3D Reasoning from Blocks to Stability. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015.

[17] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

[18] A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. In *International Conference on Machine Learning (ICML)*, pages 430–438. PMLR, 2016.

[19] T. Lozano-Pérez and L. P. Kaelbling. A constraint-based method for solving sequential manipulation planning problems. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2014.

[20] D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[21] P. Oudeyer, F. Kaplan, and A. Hafner, V.and Whyte. The playground experiment: Task-independent development of a curious robot. In *Proceedings of the AAAI Spring Symposium on Developmental Robotics*, 2005.

[22] D. Pathak, D. Gandhi, and A. Gupta. Self-supervised exploration via disagreement. In *International Conference on Machine Learning (ICML)*, pages 5062–5071. PMLR, 2019.

[23] L. G. Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

[24] Y. Shoukry, P. Nuzzo, I. Saha, A. L. Sangiovanni-Vincentelli, S. A. Seshia, G. J. Pappas, and P. Tabuada. Scalable lazy smt-based motion planning. In *IEEE Conference on Decision and Control (CDC)*, 2016.

[25] P. Shyam, W. Jaśkowski, and F. Gomez. Model-based active exploration. In *International Conference on Machine Learning (ICML)*, pages 5779–5788. PMLR, 2019.

[26] M. Toussaint and M. Lopes. Multi-bound tree search for logic-geometric programming in cooperative manipulation domains. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[27] A. M. Wells, N. T. Dantam, A. Shrivastava, and L. E. Kavraki. Learning feasibility for task and motion planning in tabletop environments. *IEEE Robotics and Automation Letters (RA-L)*, 4(2), 2019.

[28] P. Winston. The mit robot. In *Machine Intelligence*, volume 7, 1972.