IEEE International Conference on Robotics and Automation (ICRA), 2015

Learning Non-Holonomic Object Models for Mobile Manipulation

Jonathan Scholz Martin Levihn Charles L. Isbell Henrik Christensen Mike Stilman

***Abstract*— For a mobile manipulator to interact with large everyday objects, such as office tables, it is often important to have dynamic models of these objects. However, as it is infeasible to provide the robot with models for every possible object it may encounter, it is desirable that the robot can identify common object models autonomously. Existing methods for addressing this challenge are limited by being either purely kinematic, or inefficient due to a lack of physical structure. In this paper, we present a physics-based method for estimating the dynamics of common non-holonomic objects using a mobile manipulator, and demonstrate its efficiency compared to existing approaches.**

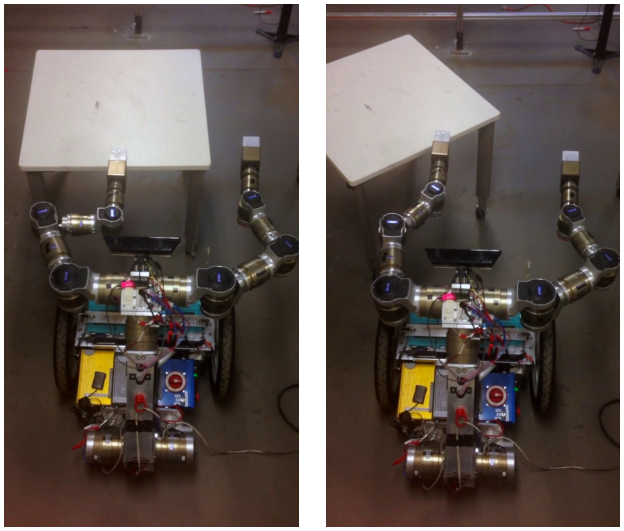
I. Introduction

One of the central challenges for mobile manipulation in natural environments is being robust to under-specified models. For example, navigating through offices and homes often requires pushing objects such as carts and chairs that have never been encountered before. Humans are able to handle such situations by quickly discovering the coarse behavior of these objects, and adapting their behavior accordingly. Imagine pushing an office table such the one depicted in Fig. 1. As soon as the table begins to rotate, humans can quickly infer that a wheel is locked and change their behavior. This ability is guided by a collection of prior beliefs about the physics of everyday objects [1]. To learn object models with only a handful of interactions, robots should utilize the same inductive bias. This paper brings robots closer to such capabilities.

To learn efficiently, it is desirable to have a model space which parameterizes common constraints, such that they can be estimated from limited manipulation data. In natural environments, the most important property governing object behavior is often physical constraints, such as hinges and wheels. To allow a robot to quickly infer these constraints, we move away from existing approaches which either require a dense training set (*e.g.* classical non-parametric methods [2, 3]), or are purely kinematic [4], and instead focus on *quickly* estimating *constrained dynamic* models. The advantage of this approach is that it is capable of representing common non-holonomic objects without resorting to non-parametric methods, allowing for model estimation with as few as one or two object interactions.

To achieve this we make the following contributions: First, we introduce Physics-Based Regression (PBR); a method adapted from [5] which is based on the idea of performing Bayesian inference over a space of constrained physical models designed for indoor environments. Next we define the necessary pipeline to convert arbitrary manipulation forces and torques (as measured using a standard wrist-mounted force-torque sensor) to the input expectations of PBR. Finally, we address the challenge of fitting approximate physical models to time-series data on a real robot, and propose optimization criteria suited for making accurate long-term predictions.

Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332, USA. Email: *jkscholz,levihn@gatech.edu, isbell,hic@cc.gatech.edu*.



(a) Start State (b) Result

Fig. 1. Mobile manipulator pushing an office table with lockable wheels. (a) Start state; robot applies force in the forward (+X) direction. (b) Result: table rotates around locked wheel.

This paper is organized as follows. Section II provides an overview of related work. Section III and Section IV detail the approach and implementation of PBR on a physical robot, which we evaluate in Section V. We provide closing remarks and discuss future work in Section VI.

II. Related Work

Within robotics, physical estimation typically focuses on inferring *mechanism* parameters, such as for a given robot manipulator [6, 7] or vehicle [8]. Physical inference of unknown objects *using* a manipulator, however, has received comparatively little attention. Two notable exceptions are [9], which develops a least-squares estimator for the full 10-DOF rigid body parametrization of a manipulator payload, and [10], which estimate a friction for a table. Both of these methods considered only unconstrained rigid-body motion, which can be formulated as a linear system.

Our work bears a close resemblance to the field of Adaptive Control, specifically indirect adaptive control [11].

Indirect adaptive control builds on the idea that a suitable controller can be determined online if a model of the plant (controlled dynamical system) is estimated online from available input-output measurements [11]. Adaptation is typically done in two stages: (1) estimation of the plant parameters using a Parameter Adaptation Algorithm (PAA) (2) updating controller parameters based on the current plant parameter estimates. PBR can be understood as a PAA generalized to support non-linear model estimation using Bayesian approximate inference. Alternative approaches to non-linear system identification can be found in [12].

Finally, there are several results in object pushing without explicit physical knowledge [13, 14], however these approaches are restricted to *holonomic* objects. We are interested in tasks in human environments, which can frequently include objects with wheels and hinges.

III. Approach

Our goal is to enable robots to be able to reliably manipulate large, unknown furniture-like objects along desired trajectories. We therefore do not assume that the robot has *a priori* models of the object dynamics, but rather needs to estimate them online based on force sensor readings and state trajectories during object manipulation. This work builds on methods from Physics-based Reinforcement Learning (PBRL), a model-based RL framework specifically designed for agents interacting with physical objects [5].

A. Physics-Based Reinforcement Learning

In this section we summarize the relevant model and inference methods used in PBRL as presented in [5]. The core purpose of the *physics-based* model used in PBRL is to capture the coarse behavior of common objects with a small number of parameters. For example, the most salient property of a shopping cart is that the rear wheels resist lateral motion along the handle axis. Less relevant are the inertial properties of those wheels, or the caster orientations in the front, as they don't qualitatively affect the control choices of the robot (for our intended applications).

PBRL utilizes this insight by defining probability distributions over the relevant physical quantities, such as masses and friction coefficients. Transitions are obtained by taking the expectation of the model's output over those random variables. With an appropriate choice of constraints, this allows us to handle a broad class of objects, while remaining tractable to estimate.

The focus of this work is on manipulation of a table with lockable wheels, which is one of the constraint types considered in PBRL. For convenience, we first review the relevant parameterization of inertia and anisotropic friction.

In the general case, inertia requires ten parameters: one for the object's mass, three for the location of the center of mass, and six for its inertia matrix I. However, if object geometry is known, we can reduce this to a single parameter m by assuming uniform distribution of mass per body fixture (*e.g.* polygon or sphere), from which I can be computed [1]. Because this paper focuses on physical constraints we utilize the scalar mass parameterization, however for a full parametrization see [9, 16].

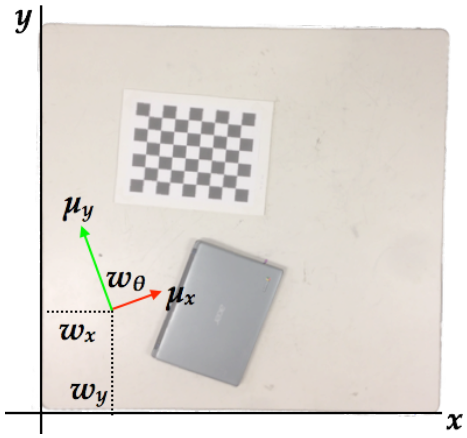


Fig. 2. Square table with wheel parameters overlay

Anisotropic friction is a velocity constraint that allows separate friction coefficients in the x and y directions, typically with one significantly larger than the other. An anisotropic friction joint is defined by the 5-vector $J_w = \langle w_x, w_y, w_\theta, \mu_x, \mu_y \rangle$, corresponding to the joint pose in the body frame, and the two orthogonal friction coefficients. Following [17], we employ a viscous-friction model to describe table motion. The instantaneous world-frame force from an anisotropic friction constraint is obtained by computing the constraint velocity in the object frame, scaling the body velocity components by their respective coefficients, and rotating back to the world frame:

$$v_{obj} = R^{-1}\left(\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & -\dot{\theta} \\ \dot{\theta} & 0 \end{bmatrix} R \begin{bmatrix} w_x \\ w_y \end{bmatrix}\right) \quad (1)$$

$$F_{wheel} = R \begin{bmatrix} \mu_x & 0 \\ 0 & \mu_y \end{bmatrix} v_{obj} \quad (2)$$

Here R is a rotation matrix describing the object's current orientation, and $\dot{x}, \dot{y}, \dot{\theta}$ represent the object's planar velocity.

Anisotropic friction constraints can be used to model wheels, tracks, and slides. One wheel is sufficient for modeling the bodies typically found indoors, such as shopping carts and wheelchairs, because they have only one constrained axis (multiple coaxial wheels can be expressed by a single constraint). The overall set of model parameters for a single body containing a single anisotropic friction constraint can be represented by the vector ϕ:

$$\phi := \langle m, w_x, w_y, w_\theta, \mu_x, \mu_y \rangle \quad (3)$$

These parameters are visualized on Fig. 2.

B. Physics-Based Regression

We now describe how PBRL can be adapted for applications on real robots. We consider object dynamics of the

[1]Mass is often parameterized in this fashion in modern simulation tools, such as Box2D [15].

following form:

$$\ddot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = I^{-1} F_{total} = I^{-1}(F_{robot} + F_{wheel}) \quad (4)$$

where $\boldsymbol{x}$ denotes the full 2D state of the object, and F_{total} the total force applied, which is decomposed into the applied robot forces $\boldsymbol{u} = F_{robot}$, and the constraint forces F_{wheel}. I^{-1} is the object's inverse inertia matrix.

Eq. 4 describes a standard second-order differential equation for table dynamics that depends on the control choices of the robot, F_{robot}, and the constraint forces, F_{wheel}, generated by the wheel(s). Any numerical integration method can be used to simulate $f(\boldsymbol{x}, \boldsymbol{u})$, but PBR preserves the use of a planar physics engine (Box2D, [15]) from PBRL to be able to handle collisions. This "greybox" approach is useful for our ultimate purpose, which is to enable robots to fit complete world-models using all the physical knowledge encoded in modern simulation tools (for further justification see [5]).

If $g(\boldsymbol{x}_i, \boldsymbol{u}_i, \delta t_i; \boldsymbol{\phi})$ denotes the discrete-time integration of Eq. 4 for time-step δt_i under model parameters $\boldsymbol{\phi}$, then the PBR regression model can be written:

$$\boldsymbol{x}'_i = g(\boldsymbol{x}_i, \boldsymbol{u}_i, \delta t_i; \boldsymbol{\phi}) + \varepsilon \quad (5)$$

where ε is zero-mean Gaussian noise with variance σ^2.

Like a standard Bayesian regression model, PBR includes uncertainty both in the process input parameters, via a prior $P(\boldsymbol{\phi})$, and in output noise ε [2]. Therefore, our target parameter space is $\{\boldsymbol{\phi}, \sigma\}$, and the target density is:

$$P(\boldsymbol{\phi}, \sigma | h) \propto P(h | \boldsymbol{\phi}, \sigma) P(\boldsymbol{\phi}) P(\sigma) \quad (6)$$

We assume that the input to the estimator is collected by a real robot and is therefore sequential:

$$h = \begin{bmatrix} \boldsymbol{x_0} & \boldsymbol{u_0} \\ \boldsymbol{x_1} & \boldsymbol{u_1} \\ \vdots & \vdots \\ \boldsymbol{x_T} & \boldsymbol{u_T} \end{bmatrix} \quad (7)$$

Estimating the PBRL model in [5] involved MCMC sampling guided by the log of Eq. 6. The (log) priors $P(\boldsymbol{\phi})$ and $P(\sigma)$ were be evaluated directly, and typically chosen to be uninformative. The log-likelihood $\log(h | \boldsymbol{\phi}, \sigma)$ was obtained by summing squared distances between the observed value and the predicted state for each state and action:

$$\ln P(h | \Phi, \sigma) \propto -\sum_{i=0}^{n} \left(\boldsymbol{x}'_i - g(\boldsymbol{x}_i, \boldsymbol{u}_i, \delta t_i; \boldsymbol{\phi})\right)^2 \quad (8)$$

A challenge when attempting to fit Eq. 8 in practice is that the trajectory data on real robots will be densely sampled, and contain both noise and un-modeled dynamical effects (*e.g.* static friction or caster orientation). In our initial experiments this lead to large data sets for relatively short manipulation episodes, as well as inaccurate long-term predictions.

[2]Priors must be chosen to restrict support to legal values. See [5] for details

To address this, PBR uses a log-likelihood term that penalizes the final integrated error, rather than incremental displacements:

$$\ln P(h | \boldsymbol{\phi}, \sigma) \propto -\left(\boldsymbol{x}_T - \boldsymbol{x}^*_T\right)^2 \quad (9)$$

where $\boldsymbol{x}^*_i$ is defined recursively as $\boldsymbol{x}^*_i = g(\boldsymbol{x}_{i-1}, \boldsymbol{u}_{i-1}, \delta t_{i-1}; \boldsymbol{\phi})$, with $\boldsymbol{x}^*_0 = \boldsymbol{x}_0$.

The main difference with Eq. 8 is that this function maintains a separate predicted state $\boldsymbol{x}^*$ over time. Therefore the actual states $\boldsymbol{x}_t$ in the trajectory are only used to evaluate the transformation of each control input $\boldsymbol{u}_i$ to the object frame (as necessary to omit grasp variables), and final error for $t = T$. This can be understood as a method for handling an under-parameterized model: By not penalizing deviations from intermediate points in the trajectory we allow greater flexibility to fit the overall displacement.

In summary, PBR can be estimated by maximizing the log-probability of Eq. 9. By contrast to [5], which uses MCMC to obtain samples from the model posterior, we utilize *L-BFGS* [18] to directly maximize the model log-probability.

IV. Implementation

In this section we define a framework that allows the approach described above to be used on real robots.

A. Prerequisites

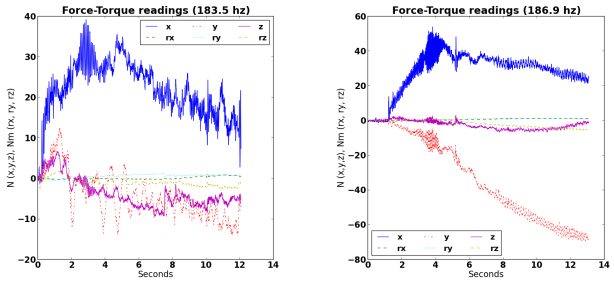
There are two basic requirements for implementing our approach on a real robot. The first is a method for measuring the reaction forces and torques during manipulation. The process presented here assumes access to measurements from a wrist-mounted force-torque sensor as well as an end-effector suitable for manipulating the target objects. We denote the raw 6-axis force-torque measurement at time t as $\mathcal{F}^{raw}_t = [\boldsymbol{F}_t, \boldsymbol{\tau}_t]^T$, which contains linear component $\boldsymbol{F} = [F_x, F_y, F_z]$ and moment $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]$.

The second requirement is a method for tracking the trajectories of objects over the course of manipulation episodes. In this work we are interested in planar-dynamics, and require the planar pose and velocity of the target object. We denote the 6-dimensional state vector containing the target object's position and velocity as $\boldsymbol{x}^O_t = [x_t, y_t, \theta_t, \dot{x}_t, \dot{y}_t, \dot{\theta}_t]$. We also require the world-frame pose of the gripper, $\boldsymbol{x}^G_t$ during the episode for mapping applied forces to the object frame, considered next. We use O_t and G_t to denote the corresponding homogeneous transforms.

B. Gathering Data

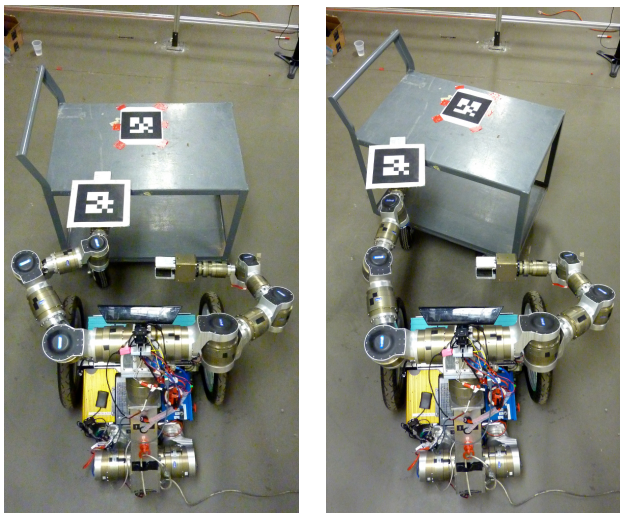
To gather data the robot applies in-plane manipulation forces to some point on the object, and records the force-torque response as well as resulting object trajectory. To avoid having to introduce the end-effector contact point in the model presented in Eq. 4, the sensor readings must be adjusted to compensate for the weight of the end-effector, and transformed to the object frame.

These are standard operations [19] which we reproduce here for completeness. First, let ${}^A_B\mathcal{T}$ denote a force-moment



(a) Table with locked caster (b) Utility Cart

Fig. 3. Raw force-torque readings from robot effector during two manipulation episodes. Robot executes a forward (+X) velocity action to (a) a table with a locked wheel, and (b) a utility cart.



(a) Start State (b) Result

Fig. 4. Humanoid robot manipulating a cart with a non-holonomic constraint. (a) Start state; robot applies force in the forward (+X) direction. (b) Result: cart rotates and slides along wheel direction.

transformation which maps force measurements from frame B to frame A:

$$\mathcal{F}^A = {}^A_B\mathcal{T}\mathcal{F}^B \tag{10}$$

$$= \begin{bmatrix} {}^A_BR & 0 \\ {}^A_BP \times {}^A_BR & {}^A_BR \end{bmatrix} \begin{bmatrix} F_0^B \\ N_0^B \end{bmatrix} \tag{11}$$

where ${}^A_BR$ and ${}^A_BP$ denote the rotation matrix and translation vector, respectively, from frame B to A.

Next, a straightforward method for gripper compensation is to cache an initial sensor offset $\mathcal{F}_0^{raw}$ and gripper pose G_0 (before object contact), and transform it to the current frame G_t using Eq. 11 at each time step:

$$\mathcal{F}_t^* = \mathcal{F}_t^{raw} - {}^{G_t}_{G_0}\mathcal{T}\mathcal{F}_0^{raw} \tag{12}$$

This reading can now be transformed to the object frame using the current transform between the gripper and the object:

$$\mathcal{F}_t^O = {}^{O_t}_{G_t}T_f\mathcal{F}_t^* \tag{13}$$

This method can handle changes in the pose of the gripper with respect to gravity, which is sufficient for our purposes, but we note that it cannot handle changes in conformation (*e.g.* finger movement) or inertial effects which may be relevant for different applications. We also note that in practice the sampling rate of the object-tracker and force-sensor may differ, so Eq. 13 should be computed using the most recent relative transforms from the object tracking system.

V. Evaluation

To evaluate the usefulness of our approach for real robotic systems, we implemented the complete framework on an actual mobile manipulator. Additionally, to obtain insight into the scalability of the approach across a range of different objects and robotic systems with different signal-to-noise ratios, we implemented the framework in realistic physical simulation.

A. Real Robot

We implemented the proposed framework on the mobile manipulator Golem Krang [20]. Golem Krang has a segway-like base with a slider in the back to provide static stability, two 7-DOF arms with 6-axis force-torque sensors in the wrist joints and 1-DOF gripper end-effectors. To obtain position estimates in the world coordinate system, we used external sensing consisting of 4 cameras tracking AR-Markers on the robot end-effector and environment objects.

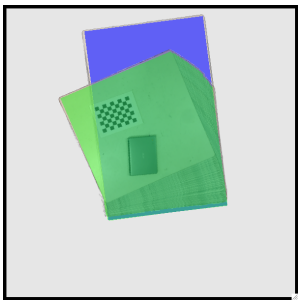
As discussed in Section IV-B, the main difference between these experiments and those in [5] is that here we consider actions as measured by a robot-mounted force-sensor that is potentially in non-rigid contact with the target object, rather than simulated forces defined at the object center-of-mass.

We performed a series of experiments using different configurations of a standard office table, and on a utility cart with fixed front wheels. The robot was tasked with applying a closed-loop velocity controlled push-action on the objects, and estimating the physical parameters of the object based on the force-torque readings as well as position estimates.

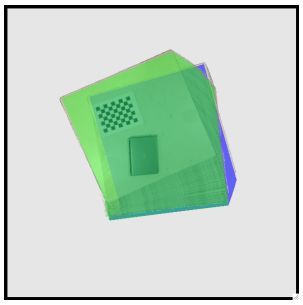
Fig. 1(a) and Fig. 1(b) show the starting and final configurations for an experiment on an office table with lockable wheels. As the robot did not know *a priori* that the lower-left table wheel was locked, the expected outcome was that it would move in a straight line. Fig. 5(a) visualizes the expected and actual obtained behavior.

The presented framework uses the observed behavior as well as the obtained force-torque readings, shown in Fig. 3(a), to estimate a model. The first attempt at model learning employed loss function utilized in Eq. 8. As can be seen in Fig. 5(b), this approach failed to produce accurate long-term predictions when applying the measured force-torque signal to the starting table configuration.

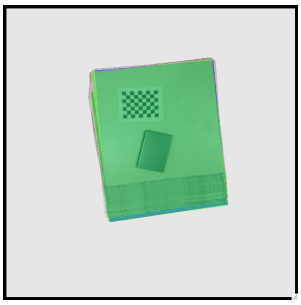
Utilizing the pipeline described in the previous section, and the integrated-error loss function Eq. 9, our method obtained a model for the object that included a wheel in the lower left quadrant of the table, with high friction in the push direction. This result consistent with the training data, but did not match our expectations, as will be discussed in



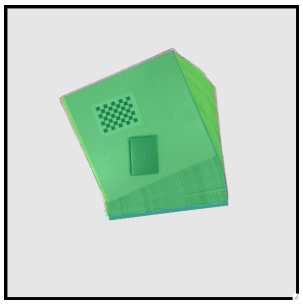
(a) Expected result: table translates



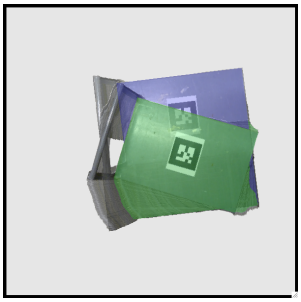
(b) Attempted learning: inappropriate loss function



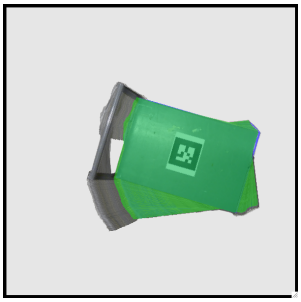
(c) Fitting unlocked table



(d) Fitting table with locked wheel



(e) Expected result: cart translates



(f) Fitting a cart with fixed wheels

Fig. 5. Overlay of observed data (green) and model predictions (blue) with and without PBR model learning. (a,e) Without learning: robot expects object to move straight forward. (b) With learning, using the full trajectory likelihood function: large error in final position estimate. (c) Learning on unlocked table: robot correctly estimates a mass and friction that reproduce the observed trajectory. (d) Learning on locked table: robot estimates that a wheel-constraint is active on the lower-left corner. (f) Learning on utility cart: robot estimates a fixed wheel on the right side of the cart.

Section VI. Fig. 5(d), visualizes an overlay of the observed table positions and a physical simulation of the estimated model when the same forces are exerted on it. In a second experiment in which all wheels were unlocked we obtained the results in Fig. 5(c), in which the mass and friction were tuned to produce the appropriate straight-line trajectory.

Fig. 5(e) shows a similar false prediction on a utility cart, resulting in the updated model shown in Fig. 5(f), containing a fixed wheel.

B. Simulation

While the previous series of experiments demonstrated the usefulness of the presented framework on a real robot, we now evaluate the scalability of the framework across a range of objects and force-torque signal-to-noise ratios. To obtain sufficient data for evaluation, these experiments were

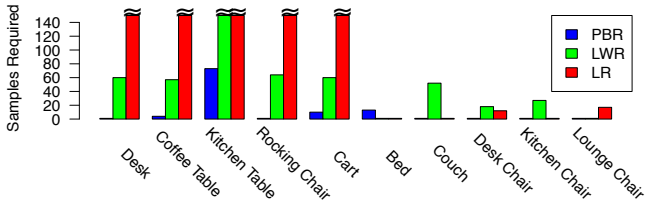


Fig. 6. Number of samples required to achieve $R^2 \geq 0.995$ on a collection of household objects

Fig. 7. Visualization of the 10 objects used for the evaluation.

performed in simulation. Further, to assess the quality of the predictions obtained by our method, we implemented Locally-Weighted Regression (LWR) [21] and Linear Regression (LR). These methods were selected because they represent common approaches at two different extrema of the bias-variance spectrum, and have complementary strengths:

- LWR is a non-parametric method with high expressive power, but requires many data points.
- LR is a parametric method which can only represent linear functions, but is very sample efficient.

Both regression methods were trained on data from Eq. 7, packaged as $x_{t+1} = f(x_t, u_t; \beta)$ for model parameters β. In LR β corresponded to the regression coefficients, and for LWR β corresponds to the Gaussian kernel hyper-parameters (for details see [5]).

1) Model Quality: Fig. 6 compares the efficiency of each model across the ten objects depicted in Fig. 7. The five left-most objects contained a nonlinear constraint in some configuration. The remaining objects were linear and only vary in shape and mass. For each object, 50 noisy training observations were gathered online and used to train a model ($\sigma = 0.50$).

For each object, bar height indicates the number of observations required by the model to obtain a given test accuracy, as measured by an r^2-statistic evaluated on 200 test samples. The samples were generated by the same procedure as used for training. The accuracy threshold was set to $r^2 \geq 0.995$. Although it may appear strict, this threshold was empirically determined such that a planner was able to solve a short (50 step) open-loop pushing task around a static obstacle with final error less than the object radius. This criterion is appropriate for the intended purpose of our model, which is to determine kinematically feasible trajectories for manipulation with velocity or force controlled

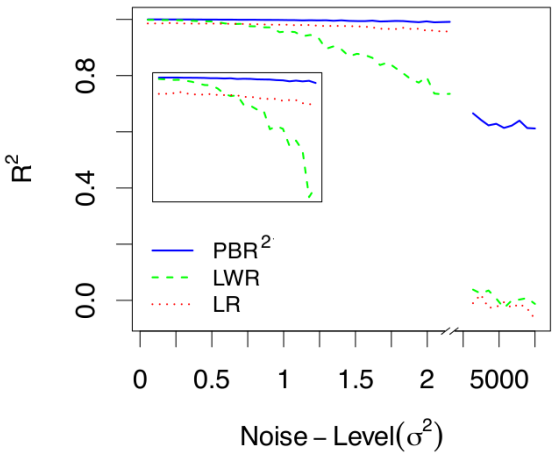


Fig. 8. Model accuracy as a function of noise level (num training = 50, num test = 200)

effectors.

As expected, LR fails to achieve the target accuracy for all five non-linear objects. However, it is efficient for the remaining ones. LWR achieves the accuracy threshold for all objects except the kitchen table. PBR was the most sample-efficient model, and reached the accuracy threshold for all objects. However, it was the worst-performing model on the bed object. Since the bed model was representable, this indicates a failure in the MCMC estimator. We plan on investigating the use of different estimators in future work.

2) Signal-To-Noise Ratio: To evaluate the generality of the proposed framework across a range of signal-to-noise ratios, we compared model variance on data gathered from a simulated shopping cart (Fig. 8). These results demonstrate that for cases in which PBR can represent the target dynamics, it is more robust to Gaussian noise than both regression methods. The inset indicates that while LWR can achieve zero-error with no noise, its performance quickly falls off as noise increases. Overall these results suggest that if an appropriate physics-based model can be defined, it can outperform more general-purpose alternatives.

VI. Conclusion

In this paper we presented a method for online estimation of non-holonomic object dynamics with a mobile manipulator. Our results suggest that anisotropic friction is a useful coarse model of wheeled object behavior which can be efficiently estimated from data available to a real robot. However, while the obtained model successfully fit the data gathered in these experiments, the estimated wheel positions didn't correspond to their actual locations on the cart and table. This could be caused by the fact that our sample space did not include all potentially relevant attributes such as static friction. It is therefore not yet clear whether the current model will be sufficiently accurate to allow appropriate control choices, such as where to grasp the object or which way to push. Future work will investigate the performance of this method in online manipulation tasks, and determine whether a richer model space is necessary for accurate planning.

Acknowledgments

This work is dedicated to the memory of Mike Stilman, whose encouragement, support and enthusiasm will never be forgotten.

References

[1] J. Hamrick, P. Battaglia, and J. Tenenbaum, "Internal physics models guide probabilistic judgments about object dynamics," in *Proceedings of the 33rd annual conference of the cognitive science society*, 2011.
[2] C. Atkeson, A. Moore, and S. Schaal, "Locally weighted learning for control," *Artificial Intelligence Review*, vol. 11, no. 1, 1997.
[3] M. Deisenroth and C. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011.
[4] R. M. Martn and O. Brock, "Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.
[5] J. Scholz, M. Levihn, C. L. Isbell, and D. Wingate, "A physics-based model prior for object-oriented mdps," in *International Conference on Machine Learning*, 2014.
[6] R. Middletone and G. Goodwin, "Adaptive computed torque control for rigid link manipulators," in *Decision and Control, 1986 25th IEEE Conference on*, vol. 25. IEEE, 1986.
[7] J. Becedas, J. Trapero, H. Sira-Ramirez, and V. Feliu-Battle, "Fast identification method to control a flexible manipulator with parameter uncertainties," in *ICRA 2007*. IEEE, 2007.
[8] K. Iagnemma, F. Genot, and S. Dubowsky, "Rapid physics-based rough-terrain rover planning with sensor and control uncertainty," in *ICRA 1999*, vol. 3. IEEE, 1999.
[9] C. Atkeson, C. An, and J. Hollerbach, "Estimation of inertial parameters of manipulator loads and links," *The International Journal of Robotics Research*, vol. 5, no. 3, 1986.
[10] M. Stilman, K. Nishiwaki, and S. Kagami, "Humanoid teleoperation for whole body manipulation," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008.
[11] I. D. Landau, *Adaptive control: algorithms, analysis and applications*. Springer, 2011.
[12] S. A. Billings, *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons, 2013.
[13] T. A. Mericli, M. Veloso, and H. L. Akin, "Achievable push-manipulation for complex passive mobile objects using past experience," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 71–78.
[14] J. Scholz and M. Stilman, "Combining motion planning and optimization for flexible robot manipulation," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*. IEEE, 2010.
[15] E. Catto, "pybox2d," http://code.google.com/p/pybox2d/, June 2012.
[16] M. Niebergall and H. Hahn, "Identification of the ten inertia parameters of a rigid body," *Nonlinear Dynamics*, vol. 13, no. 4, 1997.
[17] M. Stilman, K. Nishiwaki, and S. Kagami, "Learning object models for whole body manipulation," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*. IEEE, 2007, pp. 174–179.
[18] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization," *ACM Transactions on Mathematical Software (TOMS)*, vol. 23, no. 4, pp. 550–560, 1997.
[19] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson/Prentice Hall Upper Saddle River, NJ, USA:, 2005.
[20] M. Stilman, J. Olson, and W. Gloss, "Golem krang: Dynamically stable humanoid robot for mobile manipulation," in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 3304–3309.
[21] C. Atkeson, "Using locally weighted regression for robot learning," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*. IEEE, 1991, pp. 958–963.