A Model Predictive Approach for Online Mobile Manipulation of Nonholonomic Objects using Learned Dynamics

Roya Sabbagh Novin¹ Amir Yazdani¹ Andrew Merryweather¹ Tucker Hermans²

Abstract
A particular type of assistive robots designed for physical interaction with objects could play an important role assisting with mobility and fall prevention in healthcare facilities. Autonomous mobile manipulation presents a hurdle prior to safely using robots in real life applications. In this article, we introduce a mobile manipulation framework based on model predictive control using learned dynamics models of objects. We focus on the specific problem of manipulating legged objects such as those commonly found in healthcare environments and personal dwellings (e.g. walkers, tables, chairs). We describe a probabilistic method for autonomous learning of an approximate dynamics model for these objects. In this method, we learn dynamic parameters using a small dataset consisting of force and motion data from interactions between the robot and object. Moreover, we account for multiple manipulation strategies by formulating the manipulation planning as a mixed-integer convex optimization. The proposed framework considers the hybrid control system comprised of i) choosing which leg to grasp, and ii) control of continuous applied forces for manipulation. We formalize our algorithm based on model predictive control to compensate for modeling errors and find an optimal path to manipulate the object from one configuration to another. We show results for several objects with various wheel configurations. Simulation and physical experiments show that the obtained dynamics models are sufficiently accurate for safe and collision-free manipulation. When combined with the proposed manipulation planning algorithm, the robot successfully moves the object to a desired pose while avoiding collision.

Keywords
Autonomous Mobile Manipulation, Manipulation Planning, Service Robots

Introduction
The lack of reliable and safe mobile manipulation algorithms prevents robots from performing complicated manipulations in cluttered environments or, more importantly, in close proximity to humans, such as healthcare environments or warehouses. The ultimate goal of autonomous mobile manipulation is to perform complex manipulation tasks in dynamic environments. The manipulation task is defined as moving an object from an initial configuration to a given goal configuration (Berenson et al. (2008); Ciocarlie et al. (2012)). The reliability of the manipulation planning significantly decreases in the case of manipulating objects with unknown dynamics models. In this paper, we focus on the problem of online mobile manipulation of legged objects using learned dynamics models of objects and present an optimization-based framework for mobile manipulation planning.

One significant potential application of mobile manipulation planning is in healthcare robots. The ability for the robots to manipulate objects in healthcare environments would significantly increase the number of tasks in which robots can play a role, thus freeing professional care providers to focus on tasks that need their special expertise.

Falls resulting in injury are a prevalent patient safety problem. Every year in the United States, hundreds of thousands of patients fall in hospitals, with 30-50 percent resulting in injury. The average cost for a fall with injury is about $14,000 (Joint (2015)). Falls with serious injury are consistently among the Top 10 sentinel events reported to

¹Department of Mechanical Engineering and Utah Robotics Center, University of Utah, Utah, US
²School of Computing and Utah Robotics Center, University of Utah, Utah, USA

Corresponding author:
Roya Sabbagh Novin, Department of Mechanical Engineering, University of Utah, Utah, US.
Email: roya.sabbaghnovin@utah.edu
The Joint Commissions Sentinel Event database, with the majority of these falls occurring in hospitals (Chu (2017)). One of the most important healthcare factors related to falls is the nurse-to-patient ratio (Chu (2017)). Studies show that there is a significant relationship between falls and short staffing/nurse workload (Chu (2017)). In these situations, even when an alarm activates, it may take minutes for a nurse to respond and falls often happen during this response time period when patients have little to no support especially while ambulating to and from the bedside to the bathroom (Kristoffersson et al. (2013); Oliver et al. (1997)). Patient sitters are one of the solutions to overcome this problem. However, a patient sitter could be replaced with assistive robots.

We believe an autonomous assistant mobile robot with object manipulation capabilities can help to prevent patient falls by intervening with a mobility aid at the bedside. The robot uses monitoring data to plan assistance by providing a mobility aid to a patient, or clears the patient’s path by moving obstacles away (Fig. 1).

One of the main challenges in using robots for such applications is safety. A robot maneuvering in close proximity to humans needs to consider human motion and intention to avoid any collision. The problem of human-aware autonomous mobile robot navigation in cluttered and dynamic environments has been widely investigated (Nakhaeinia et al. (2015); Kretzschmar et al. (2016); Pol and Murugan (2015)). However, many challenges still exist in manipulating objects while navigating through cluttered and unstructured environments such as hospitals or personal dwellings (Nakhaeinia et al. (2015); Kretzschmar et al. (2016); Pol and Murugan (2015); Mast et al. (2015)). This includes estimating the dynamics of unknown objects, creating safe and collision-free maneuvering trajectories and dealing with discrete and continuous, i.e. hybrid actions. Medical environments are usually cluttered by various objects, including mobility aids, carts, chairs and tables. Since the dynamics of these objects are not necessarily known to the robot, the robot must be able to autonomously determine the object’s dynamics properties for successful manipulation. Additionally, when manipulating legged objects, the robot must select not only a direction and magnitude of the pushing or pulling force, but also make a discrete choice of which leg to push or pull. Thus, in this paper, we investigate the problem of manipulating unknown legged objects to a desired final position using our customized robot (Fig. 2).

Previous research concerning dynamics models mostly describe either mapping between actions and consequences for a specific task (Ogata et al. (2005); Fitzpatrick et al. (2003)), or rely on pure kinematics (Vithani and Gupta (2002)). A more advanced approach mimics human sensorimotor learning behavior, in which a coarse dynamics model of the new object is learned based upon prior beliefs and experiences. Eventually, the coarse model is improved as more data are collected during the manipulation (Scholz et al. (2015); Körding and Wolpert (2004)). In this research we choose a Bayesian regression model in order to incorporate knowledge about common legged furniture as priors to inform the dynamics learning algorithm (Körding and Wolpert (2004)).

For autonomous manipulation we develop a model predictive controller (MPC) based on mixed-integer convex optimization to overcome the imperfections of the dynamics model and avoid getting stuck in local minima. To make it convex, we linearize the dynamics model over a nominal trajectory for the object. We incorporate the hybrid actions by penalizing actions that require changing legs based on the path between legs and costs associated with regrasping.

We divide the mobile manipulation problem into three sub-problems: 1) move the robot from its initial state to a state where it is near the object and can move to a grasping state, 2) move to a grasping state and grasp, 3) move the robot (grasping the object) in such a way that it moves the object into its goal configuration.

Below, we summarize the main contributions of this paper:

1. Unknown Object Dynamics Learning: Using Bayesian regression method adopted from Scholz et al. (2016) to learn dynamic parameters of legged objects and investigate three different dynamics models using experimental data.

2. Hybrid Manipulation Planner: Development of a manipulation planner based on receding horizon concept and mixed-integer convex optimization with discrete actions of changing legs as well as continuous motion in manipulation.

This paper builds on the preliminary results presented in the conference paper (Sabbagh Novin et al. (2018)), where the robot design, the procedure for object parameter estimation and main concepts of manipulation planning algorithm were introduced. The major improvements with respect to this previous work are:

1. We introduce an additional mode for grasping, which reduces multiple efforts to grasp a leg and makes overall object manipulation faster.

2. We develop a repositioning mode, which prevents the robot from getting stuck in between two legs of the object.

3. We investigate more complicated dynamics models and run the dynamics model learning on several objects.

4. We use a new system of weight assignment for the optimization cost function.

5. We perform simulation and physical experiments using the proposed manipulation planning algorithm on multiple legged objects.

We organize the remainder of the paper as follows. We discuss an overview of related work in Section 2. In Section 3, we introduce and formalize the mobile manipulation problem. We follow this with details of our approach in Section 4, including the dynamics parameter learning method and manipulation planning algorithm. In Section 5, we explain three different dynamics models and the...
implementation aspects of Bayesian regression on collected data using a real robot and the experimental protocol for evaluating our object manipulation algorithm. We analyze the results of extensive robot experiments in Section 6. Finally, a closing discussion and potential future work are presented in Section 7.

Related Work

Assistive robots

Most assistive robots developed to support independent living are only used to monitor, communicate or deliver supplies in hospitals without any physical engagement with patients (e.g. Giraff by Pripfl et al. (2016) and HOBBIT by Casiddu et al. (2015)). However, these robots have the potential to do more interactive tasks that are repetitive, time consuming and do not need the expertise of a professional care provider. Chen et al. (2013) have developed assistive capabilities for the PR2 robot to empower people with severe motor impairments to interact with the physical world. They have investigated a range of tasks through two case studies including scratching and shaving, retrieving an object at home, and socially interacting through speech and gesture. In all these tasks, the robot is directly controlled by the human.

Object model identification in motion planning

The estimation of dynamic parameters of a manipulated object, by autonomous mobile robot, has received some attention in the past. Most of the existing approaches either require a large training dataset (Fitzpatrick et al. (2003)), or use kinematics-based methods for a specific task (Vithani and Gupta (2002)).

Some studies are based on learning a mapping between actions and the resulting effects to describe an object’s dynamic behavior and inform future goal-directed behavior (Ogata et al. (2005); Fitzpatrick et al. (2003)). The most limiting drawback in these methods is that since they do not provide any physics-based dynamics model, they cannot be used for other types of manipulation other than what is performed in the training process. Thus, they are very limited in terms of handling new task requirements.

To overcome the limitations of mapping methods, some studies suggest finding dynamic parameters of objects instead. Stillman et al. (2007) used the pseudo-inverse of dynamics equations to obtain the dynamic parameters of large objects modeled as “a point mass on a wheel”. However, they were not successful in finding a consistent relationship between acceleration and force and only used a viscous friction model ignoring mass and inertia parameters.

Later, learning methods were used to estimate non-linear dynamics models of objects. Scholz et al. (2015) used physics-based reinforcement learning as an adaptive method to obtain dynamics models of nonholonomic objects. They used this method to estimate the physical parameters of an office table and a utility cart with fixed front wheels using the same “point mass on a wheel” model (Scholz et al. (2016)).

Least squares approaches are also used for object kinematic and dynamic parameter estimation (Cehajic et al. (2017)). Some literature use interaction data between a team of mobile robots and object to find mass and inertia parameters (Marino and Pierri (2018); Franchi et al. (2015)). More complicated models are also investigated for nonholonomic objects. Sun et al. (2002) use least squares to identify model parameters of a 4-wheel cart manipulated by a mobile manipulator. However, they ignore friction effects.

Manipulation planning

There is also an ongoing effort to find planning frameworks that can effectively handle the uncertainty and hybridness associated with planning for both pushing and pulling actions. Mason (1986) first formulated the mechanics of planar pushing manipulation tasks. Salganicoff et al. (1993) created a forward empirical model of an unknown object for pushing using visual feedback. Li and Payandeh (2007) focused on finding appropriate pushing actions and developing a push planner which can track a predefined trajectory using these actions based on a set of assumptions and a simplified model of two-agent point-contact push.

Arruda et al. (2017) used a model predictive path integral controller to plan push manipulations based on a learned model including uncertainties, obtained by Gaussian process regression and an ensemble of mixture density networks. Hermans et al. presented a data-driven approach for learning good contact locations for pushing unknown objects (Hermans et al. (2013)).

Desai and Kumar (1997) addresses the problem of motion planning for nonholonomic cooperating mobile robots manipulating and transporting objects while holding them in a stable grasp. They use the calculus of variations (with high computational cost) to obtain optimal trajectories and actuator forces and torques for object manipulation in the presence of obstacles. In their planning scheme, they only plan for the pushing action, assuming that robots have already grasped the object and do not need to plan for the grasping position.

A few model-based hybrid manipulation controllers have been introduced (Woodruff and Lynch (2017); Hogan et al. (2017); Hogan and Rodriguez (2016)). The control strategies presented in the aforementioned papers are applied to systems with a priori knowledge of the contact mode sequencing or offline determination of optimal mode sequences. In Hogan et al. (2017), MPC is used to find an optimal sequence of robot motions to achieve a desired object motion. Pajarinen et al. (2017) solves the problem of finding an optimal sequence of hybrid controls under uncertainty using differential dynamic programming and incorporating discrete actions inside DDP.

Problem Statement

The problem of mobile manipulation is defined as controlling a dynamical system

\[ \dot{x} = f(x, u) \] (1)

such that it takes the system to a desired state. For solving this problem, we consider the discrete form of the dynamics which is an approximation of real dynamics of the system:

\[ x_{t+1} = \hat{f}(x_t, u_t) \] (2)

With \( x_t \) and \( u_t \) representing system state and control input, respectively, at time \( t \).
In this paper, we use an MPC approach to find the desired control input. At each time step, we find the optimal control sequence \( u = \{u_{t+1}, ..., u_{t+H}\} \) for a limited horizon \( H \) following the approximated dynamics model \( f \) resulting in a sequence of system states \( x = \{x_{t,1}, x_{t,2}, ..., x_{t,H}\} \). We apply the first control input in the sequence \( u_{t+1} \), and replan for the next step, until it reaches a state in the goal set. Through this replanning framework, we desire to minimize an optimal control objective, which in a mobile manipulation problem consists of two main components of the system, the robot and the object. In other words, we need to find the state control sequence which solves

\[
\min_{x,u} J(x, u) \tag{3}
\]

with respect to the constraints of the problem which includes obstacle avoidance, as well as kinematics and dynamics constraints. Once this optimization is solved, we extract the first control input and apply it to the real system \( f \). Since, at each time step, we replan based on the real current state \( x_t \), the entire controller behaves as a state-feedback controller and is able to compensate for local model and perception errors. The objective function is defined as the performance of the robot and the object:

\[
J(x, u) = c_R(x, u) + c_D(x, u) \tag{4}
\]

\( c_R \) represents all the costs associated with robot’s motion and \( c_D \) is defined as the overall cost for the object. Each cost function consists of a cost over the entire horizon \( c_P \), and a terminal cost considering the last state in the horizon \( c_T \).

\[
c_i(x, u) = \sum_{h=1}^{H-1} c_{i,P}(x_{t,h}, u_{t,h}) + c_{i,T}(x_{t,H}, u_{t,H}) \tag{5}
\]

We will define these specific costs for the application of legged objects manipulation. In the following sections, we discuss the details of how we find an approximation for the system dynamics and the manipulation planning framework using MPC.

**Dynamics Models of Objects**

For object motion estimation, we prefer learning the dynamics model since our task (i.e., which object to move) is not defined a priori. We use Bayesian regression to predict dynamic parameters from observed motion and force data. For this purpose, we consider three different models: (1) a simple model of point mass on a wheel, (2) a 2-wheel model, (3) a friction-only model. These models are shown in Fig. 3. The method we use here is adopted from Scholz et al. (2015). The obtained model will provide us with a probabilistic estimate of the dynamic parameters of the object for a given model.

Input data includes force data and the resulting object motion. We only apply force and assume the applied torque is zero. The reason to avoid collecting torque data is to develop a simple and general model with a small dataset. However, the algorithm could be implemented with a torque sensor as well.

We only consider planar parameters since the objects of interest will only slide or roll on the floor. Dynamic parameters for a planar model include inertia and friction. For the first two models, inertia requires four parameters for planar manipulation: one for mass, one for inertia in the X and Y plane and two for the center of mass position. The difference in these two models comes from the friction. For the point mass on a wheel we define two friction coefficients \( \mu_x \) and \( \mu_y \) in the X and Y directions to define the anisotropic friction and \( \theta_\mu \) for the wheel orientation. In this case, the model parameter vector is:

\[
\Phi_1 := \langle m, I, x_c, y_c, \mu_x, \mu_y, \theta_\mu \rangle \tag{6}
\]

However, for the second model, we have two wheels, resulting in 4 friction coefficients, \( (\mu_{x,l}, \mu_{y,l}) \) for the right wheel and \( (\mu_{x,r}, \mu_{y,r}) \) for the left one. But we assume the orientation of the wheels are known. Another set of important parameters are the position of wheels which is defined by a center of wheel shaft position \( x_s \) and \( y_s \) and the distance between two wheels \( b \). The parameter vector for this model is:

\[
\Phi_2 := \langle m, I, x_c, y_c, \mu_{x,l}, \mu_{y,l}, \mu_{x,r}, \mu_{y,r}, x_s, y_s, b \rangle \tag{7}
\]

In the third model, we investigate the effect of inertia by only considering the friction parameters for the 1-wheel model. For this we introduce a new friction term \( \mu_\theta \) to represent resistance to rotation, resulting in four friction coefficients in total. The parameter vector for this model is:

\[
\Phi_3 := \langle x_c, y_c, \mu_{x,l}, \mu_{y,l}, \mu_\theta, \theta_\mu \rangle \tag{8}
\]

We use Bayesian regression to find these parameters which are presented as random variables from a prior probability distribution in the model. Then, we find the conditional probability of possible values of these random variables based on the given observation. Since the posterior distribution cannot be reasonably obtained by direct computation, we use a Markov Chain Monte Carlo (MCMC) method to sample from the distribution (Bernardo and Smith (2001)). We define physics-based prior distributions for dynamic parameters and present them as a truncated normal distribution with mean value \( \mu \) and standard deviation \( \sigma \) since all these parameters have lower and upper bounds \((l_i, u_i)\):

\[
\Phi \sim N_l(\mu, \sigma, l_i, u_i)
\]

Next, we derive the dynamics equations. To obtain the friction force for each wheel in the models, we need to compute the velocity of the wheel in the wheel frame, find the friction force components, and finally convert it back to the world frame. Using the second model, the wheel frame and object frame are the same as the orientation of the wheels are fixed. The velocity of the wheel in the wheel frame is computed based on the center of mass velocity as:

\[
v_w = R^{R-1} R^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} R \begin{bmatrix} x_w \\ y_w \end{bmatrix} \tag{9}
\]

\[
F_w = R R' \begin{bmatrix} \mu_{x} \\ 0 \\ \mu_{y} \end{bmatrix} v_w \tag{10}
\]

In the above, \( R \) is a rotation matrix from the world frame to the object frame and \( R' \) is a rotation matrix from the object frame to the wheel frame. \([\dot{x}, \dot{y}, \dot{\theta}]\) represent the objects
planar velocity and $[x_w, y_w]$ are position of the wheel in the object frame. In the 1-wheel model, this is the same as the center of mass position. However, in the second model, the position of wheels are defined using the center of wheel shaft.

$$ x_w = x_s \pm b, \quad y_w = y_s $$

(11)

The total force/torque $F_T$ is the sum of all friction forces/torque ($F_{w,i}$, $\tau_{w,i}$) and the input force/torque ($F_r$, $\tau_r$) which results in the following dynamics:

$$ F_T = \begin{bmatrix} F \\ \tau \end{bmatrix} = \begin{bmatrix} F_r + \sum F_{w,i} \\ \tau_r + \sum (R_{w,i} \times F_{w,i}) \end{bmatrix} $$

(12)

$$ \ddot{x} = I^{-1} F_T $$

(13)

Where $I$ denotes the object’s inertia matrix. Taking the input as the applied force by the robot ($u = F_r$) and reforming Eq. 2 as a discrete time integration over time steps $\delta t_i$, and additive Gaussian noise $\epsilon_i$ with zero mean and variance $\sigma^2$ results in:

$$ x_{i+1} = f(x_i, u_i, \delta t_i, \Phi) + \epsilon_i $$

(14)

Which defines our Bayesian regression model. Both input variables and output noise include uncertainty. We find the probability of dynamic parameters and output noise using the input dataset $D = \{x, u\}$ and Bayes theorem:

$$ P(\Phi, \sigma|D) \propto P(D|\Phi, \sigma) P(\Phi) P(\sigma) $$

(15)

**Mobile Manipulation Framework**

In this section, we discuss various parts of our manipulation planning framework. The manipulation planning algorithm is divided into 3 major modes: (1) motion planning from robot’s initial position to the object grasping position, (2) grasping mode to grasp the object’s leg, and (3) manipulation planning to move the object to the desired state. To define the MPC objective function we have to consider all three modes.

In the first two modes, the object is stationary, so we only define cost for the robot motion. In the third mode, both the robot and the object are involved, however, when the object is grasped, it moves with the robot. Thus, minimizing the cost for the robot or the object, also minimizes the cost of the whole system.

For this mode, since the smoothness of the object path is more important for us, we define the cost for the third mode as the path length of the object. At each time step, we assume that the robot is not changing legs during the current horizon. However, we know that this may not necessarily be true because the robot can get stuck between two legs of the object. In this case, replanning and repositioning is necessary.

To find the cost of repositioning, after planning optimization, we simulate the planned trajectory from optimization with the approximate dynamics. Then, we find the cost by counting the number of repositioning actions needed for that plan and add it to the total cost:

$$ c_T = c_{\text{motion}} + c_{\text{grasp}} + c_{\text{manipulation}} + c_{\text{reposition}} $$

(16)

In the following, we discuss how we calculate each of these costs and choose the optimal leg for manipulation. At each time step, we calculate the total cost for each leg of the object and choose the one with minimum total cost. Algorithm 1 provides the pseudocode of the hybrid manipulation planning function (”opt”). The inputs to this function are the target object’s current and goal states $(\xi_s, \xi_g)$, robot’s current state $\xi_r$ and all other objects which are considered as obstacles $(O)$.

For each leg of the object, first, we find a manipulation plan assuming that the robot has grasped that leg ($c_{\text{manipulation}}$). However, since the dynamic constraints are not linear and cannot be used in the convex optimization directly, we find a nominal trajectory based on the object’s current and desired state (Line 4) and linearize the dynamic constraints over the nominal trajectory. We use a simple version of a path planning problem with obstacle avoidance but without dynamics constraints for a horizon length that...
Algorithm 1: Hybrid manipulation planner used in “opt”

Result: $\pi^*$

input : $\xi_o, \xi^0_T, \xi_r, O$

(1) $c_{\text{min}} \leftarrow \infty$
(2) $\pi^* \leftarrow \emptyset$
(3) for $i$ in legs do
(4) $T_n \leftarrow \text{nominal_traj}(\xi_o, \xi^0_T, O)$
(5) $c_{\text{manipulation}} \leftarrow \text{opt_manipulate}(\xi_o, \xi^0_T, O, T_n)$
(6) $c_{\text{motion}} \leftarrow \text{opt_motion}(\xi_r, \pi, O)$
(7) $c_{\text{reposition}} \leftarrow \text{simulate}(\pi)$
(8) $c_{\text{grasp}} \leftarrow \text{change_leg}(l_i, l)$
(9) $c_T \leftarrow c_{\text{manipulation}} + c_{\text{motion}} + c_{\text{reposition}} + c_{\text{grasp}}$
(10) if $c_T < c_{\text{min}}$ then
(11) $c_{\text{min}} \leftarrow c_T$
(12) $\pi^* \leftarrow \pi$
(13) end
(14) return $\pi^*$

is the the same as the main problem to find the nominal trajectory.

Based on the obtained manipulation plan, we find a grasping goal for the robot and find the cost of the robot motion to get to the grasping goal ($c_{\text{motion}}$) which is calculated in the “opt_motion” function. If there is a need for a new grasp, we add grasping cost ($c_{\text{grasp}}$) which is the cost of releasing the current leg and grasping another one. Otherwise, if the plan is for the current grasp, we assign zero grasping cost. In order to find the cost for repositioning, since this is not directly included in the optimization, we simulate the entire manipulation plan and count the number of repositioning actions required for that plan and use that to calculate the cost of repositioning ($c_{\text{reposition}}$). Combining all four costs gives us the cost for one leg. We repeat this procedure for all legs and find the minimum cost leg.

If the minimum cost is infinity, it means a feasible trajectory was not found, so the robot stays put. Otherwise, the “opt” function returns the optimal plan $\pi^*$. It should be mentioned that we scale all the weights in such a way that the robot avoids changing legs when far from the desired configuration and gradually decrease the weight for grasping to allow final tuning of the object’s configuration.

Once the minimum cost plan is chosen, the current mode of the robot is found based on the distance of the robot to the desired leg and also the distance of the walker to the desired configuration. Algorithm 2 presents the high-level framework which controls the modes and sends commands to the low-level controller of the robot according to the current planning mode. The inputs to the algorithm are: object current state $\xi_o$, object goal state $\xi^0_T$, robot current state $\xi_r$, set of obstacles $O$ and dynamics parameters $\Phi$. The goal zone for the object and the robot are defined by $\epsilon$ and $\epsilon_R$.

At each time step, while the object is not in the goal region ($||\xi_o - \xi^0_T|| > \epsilon$), if there is no valid plan or there is any change in the environment including the object’s state, we run the complete planning optimization using the “opt” function (line 6,7). If it finds a feasible plan, it goes to one of the four modes mentioned above:

1. If the robot is far from the walker, it is in motion planning mode (lines 22-23) and we use motion planning optimization function “opt_motion” to find the optimal plan.
2. After it reaches the grasping zone ($||\xi_r - \xi^0_T|| < \epsilon_R$), it goes into the grasp mode until it successfully grasps the object’s leg using a pre-defined grasp plan (lines 19-20).
3. Then, the manipulation mode starts (line 16-17) and replans the manipulation at each step using “opt_manipulation” function. It will repeat this until the object is in the goal zone.
4. Whenever the robot is close to getting stuck between two legs of the object, it will go to repositioning mode (lines 13-14) and continue from a new direction (lines 2-3). This is only for the situation in which the current leg is still the best leg for manipulation and only the position of the robot is not favorable.

At each step, based on the feedback, the robot can decide that continuing with another leg has less estimated total cost. In that case, it will release the grasped leg and do the motion planning and the grasping part all over again for the new leg. The cost of this procedure is considered in the total cost, therefore, this will not happen unless the manipulation cost is significantly improved by changing legs, or finishing the task with only one leg is not feasible at all.

In the next sections, we explain structures of different modes in the algorithm.

Algorithm 2: High Level Controller

Result: Next action

input : $\xi_o, \xi^0_T, \xi_r, O, \Phi$

while $||\xi_o - \xi^0_T|| > \epsilon$ do

if mode $\pi$ “re - positioning” then

$\pi \leftarrow \text{move_to_robot_goal}(\xi_r, \xi^0_T, O)$

else

if change or $\pi = \emptyset$ then

$\pi, l^*, \xi^0_T \leftarrow \text{opt}(\xi_o, \xi^0_T, \xi_r, O, \Phi)$

$l_c \leftarrow \text{find_current_leg}(\xi_o, \xi_r)$

if $\pi \neq \emptyset$ then

if $l_c = l^*$ then

out_of_limits $\leftarrow \text{check_limits}(\xi_o, \xi_r)$

if out_of_limits then

mode $\leftarrow \text{re-positioning}$

$\xi^0_T \leftarrow \text{find_new_goal}(\xi_r, \pi)$

else

mode $\leftarrow \text{manipulation}$

$\pi \leftarrow \text{opt_manipulate}(\xi_o, \xi^0_T, O, \Phi)$

else if $||\xi_r - \xi^0_T|| < \epsilon_R$ then

mode $\leftarrow \text{grasping}$

$\pi \leftarrow \text{grasp_plan}(\xi_r, \xi^0_T, O)$

else

mode $\leftarrow \text{robot motion}$

$\pi \leftarrow \text{opt_motion}(\xi_r, \xi^0_T, O)$

if $\pi = \emptyset$ then

return StayPut

else

return $\pi[0]$

end
Motion and manipulation modes

The motion planning and manipulation planning modes share the same core optimization structure, with some additional constraints for the manipulation mode to incorporate the dynamics of object manipulation. We define the planning problem as an MPC-based optimization to obtain an optimal path from initial configuration to the desired configuration.

We formulate our planner as a mixed-integer convex optimization problem (Boyd and Vandenberghe (2004)), which is defined as a general optimization problem with convex objective function $J(x, u)$ and convex inequality functions $g_i(x, u)$ or piecewise affine equality functions $f_i(x, u)$ as constraints:

$$\min_{x} J(x, u)$$

s.t. $f_i(x, u) = 0$, $i = 1, \ldots, m$

$$g_j(x, u) < 0$, $j = 1, \ldots, n.$$

In the following, we provide details on all component of the optimization problem used both in motion planning for the robot and manipulation planning of the object.

Cost function:

Since we are using convex optimization framework, the cost function must have a convex form. Here, we define it as a shortest path cost function with the purpose of finding a smooth trajectory around obstacles and furniture and avoiding unnecessary motion. At this point, we are not concerned about time because the velocity limits of the robot will not allow for fast motion. However, with a more powerful robot, having a combination of minimum time and shortest path would be a better option. By considering a control horizon with length $H$, we can write the cost function for the shortest path as:

$$c = \omega_1 \sum_{h=0}^{H-1} \delta p + \omega_2 \delta T$$

(17)

where $\delta p = ||\xi_{t+h+1} - \xi_{t+h}||^2_2$ is the change in the object’s state between two steps in the horizon and $\delta T = ||\dot{\xi}(t, H) - \dot{\xi}(s, 0)||_2$ is the terminal cost and shows the difference between the final state in the horizon and the goal state. $(\omega_1, \omega_2)$ are weight matrices to adjust based on the importance of each term in the cost function.

It should be mentioned that in the manipulation problem we define this based on the object’s state. However, for the motion planning mode which only includes the robot’s motion, we use the robot’s states to define the cost function. Moreover, in manipulation planning, we use different values for the weights on position and orientation costs.

We use lower ratios of orientation cost over position cost when the object is far from the goal state and increase it as the object gets closer to the goal state. This is mainly because in our application the orientation of the object in the middle of the trajectory is not as important as getting the object to the goal position. So, here $(\omega_1, \omega_2)$ change over time during one task.

Obstacle avoidance:

The obstacles are written as equivalent surrounding convex forms. Therefore, each obstacle is approximated by a polygonal shape. Although this is slightly more conservative than a point cloud, since most of the objects in medical environments are box shaped like hospital beds or chairs, this assumption does not have a dramatic effect on the optimality of the solution. Polygon shapes are defined as the intersection of a series of half spaces:

$$O : \{ \xi | A \xi < b \}$$

(18)

The point $\xi$ is outside of shape $O$ with $m$ number of sides if at least one of the $A \xi < b$ inequalities is satisfied:

$$A \xi < b + (v - 1)M, \sum_{i=1}^{m} v_i \geq 1$$

(19)

where $v$ is a vector of binary variables and $M$ is a large constant value used in the Big-M method (Richards and How (2005)). Equation 19 ensures that at least one element of the vector $v$ equals to 1, so point $\xi$ would be out of the polygon obstacle.

Kinematics constraints:

The kinematic constraints include initial state constraint and velocity/acceleration limits of the objects or the robot. Considering equal timesteps $dT$, these constraints can be formulated as below:

$$\xi(0) = \xi_s$$

(20)

$$\frac{\xi(h+1) - \xi(h)}{dT} \leq \dot{\xi}_{max}, \quad h = 0, \ldots, H$$

(21)

Dynamic constraints:

Finally, a set of constraints that change the motion planning problem to the manipulation problem is the set of dynamics constraints. Dynamic constraints play the main role in the optimization problem for manipulation and connects the applied force to the object’s motion. Dynamic equations which are the result of the dynamics learning discussed earlier, define the relationship between the applied force by the robot and the resulting trajectory of the object.

These constraints are non-linear and should be linearized before we can use them in the convex optimization problem. We do this by finding an approximated nominal trajectory and using it to linearize the dynamics equations. This would add errors to the solution, but implementing it in a receding horizon framework will compensate for it.

$$\xi(h+1) = f(\xi(h), u(h)), \quad h = 0, \ldots, H$$

(22)

Another part of dynamics constraints are the limitations on the magnitude and direction of force.

$$|u(h)| \leq u_{max}, \quad h = 0, \ldots, H$$

(23)

To conclude, the whole optimization problem is provided below which is used in both the opt_manipulate and opt_motion functions.

$$\min_u J = \omega_1 \sum_{h=0}^{H-1} ||\xi(h+1) - \xi(h)||^2 + \omega_2 ||\xi(H) - \xi_0||^2$$
We change the cost of grasping based on the distance to the goal state. When the robot is far from the goal, we have high regrasping cost to avoid changing legs as much as possible; but, when the robot gets closer, we want to have the flexibility of using different legs to adjust the object’s orientation as desired.

Grasping mode

For the grasping mode, we use a predefined motion primitive to guarantee that the robot can grasp the desired leg of the object. This is mainly because of the robot’s limitation in performing fine motion plans. For example, the resolution of rotation is not accurate enough for final adjustments before grasping, or it would take a long time for the robot to perform a lateral transition due to its non-holonomic behavior.

We avoid this by repositioning whenever the robot gets too close to the object’s side, which can lead to getting stuck between legs. After repositioning, the robot will replan and continue manipulation. The repositioning mode, includes releasing the leg, moving to the new grasping goal which is obtained by the desired force direction, and re-grasping. This process and its solution are presented in Fig. 4.

The cost of repositioning is calculated based on the number of repositioning actions needed to perform the entire plan. We do this as a secondary step by simulating the plan using the dynamics model in a manner the same as the one used in the optimization. Then, we scale it by estimating the average cost of moving from one leg to another.

Experiments

For the experimental studies, we use a low-cost mobile robot based on an iRobot Create2 with a customized 3-finger gripper which was introduced in Sabbagh Novin et al. (2018).

We collect synchronized data for dynamics model learning using a motion capture system with 20 Flex13 cameras (Optitrack, Naturalpoint, Inc.) and a one-directional force sensor mounted on the robot’s gripper (Futek Industries). We implement our approach on four different objects; a 2-wheel walker, two 4-wheel chairs and a 4-wheel rack.

For each object we collect data from about 70 short trajectories and divide them to create a 50 element training dataset and a 20 element test dataset. Each trial is about 5-15 seconds and is collected at a 10Hz sampling rate. Force data are filtered by a 6th order butterworth filter to smooth the noisy input. In each trial, the robot pushes or pulls one of the object’s legs starting from one of the possible directions. We assume that the robot’s gripper acts like a revolute joint and can only apply force and no torque is applied to the object’s leg. It should be mentioned that for some objects, the robot could not grasp all the legs due to the shape of the leg or extra support bars between legs. In those cases we only collect data from the legs that are possible for the robot to grasp.

For the Bayesian regression model, we have used the PYMC package (Patil et al. 2010) in python with 20000 samples running on a Core i7 2.4GHz system. We run model learning for each object using all three models for comparison.

For the simulated manipulation planning experiments, we use the same objects with the learned dynamics models in our simulation setup and run our proposed method to manipulate objects from the initial state to a desired state. We define 4 different scenarios with various initial states of the object and the robot, the desired state for the object and room configuration. For each object, we run each scenario 50 times and report the success rate, position and orientation error and average run time. A trial is successful if the robot can take the object to the goal region in less that 3
Figure 5. Final displacement errors with and without feedback for three different models: (1) Point mass on a wheel model, (2) 2-wheel model, (3) Friction-only model. The displacement error is defined as the difference between predicted and actual displacement at the end of the trajectory.

Results

In this section, first we discuss final results for the dynamics model learning, showing the errors for all three types of dynamics models. Then we perform a thorough evaluation of our proposed mobile manipulation planning framework, in simulation and physical experiments.

Dynamics model learning

As previously stated, the object’s dynamics model is learned using MCMC sampling. On a Core i7 2.4GHz system, it takes about 1 hour to get 20000 samples.

After obtaining an object model based on the training dataset, we tested it on our test dataset containing about 20 trajectory episodes. Each trajectory prediction begins from the actual starting point and then we only use the actual dataset as feedback input every 2 seconds.

For better evaluation of the model types, for each model, a plot of final displacement errors with and without feedback, which is the difference between predicted and actual displacement at the end of the trajectory is presented in Fig. 5. It is shown that including inertia parameters improves prediction significantly. In addition although the second model works slightly better for the walker, the difference is not significant. As a result, since a more complex model results in higher computational time in optimization, we choose the first model which is simpler than the second model.

Figure 6 provides a comparison between the predicted trajectory without feedback, predicted trajectory with feedback and the actual trajectory using the first model. As expected, we can see that using feedback helps to stay on the trajectory and eliminate the accumulated error every two seconds, however, it does not change the displacement error much since this is based on the obtained model and noise in the system.

Additionally, we can see that in the far left case, errors get higher as we move towards the end of the trajectory. We believe this is due to forces applied by the gripper fingers which are not measured in this study. This happens in cases when the force direction is such that gripper fingers apply more force and actually play a role in influencing dynamics. A better force and torque measurement approach is needed to get more accurate results.

Manipulation planning

We ran the manipulation planner for four different simulation setups for all objects (Fig. 7). For each simulation trial, the actual dynamic parameter is sampled from the learned distribution. However, in the planning, we always use the mean value. We also add noise to the system for simulating the resulting trajectory. For each setup, we compare results of 50 trials from our approach and LQR using an initial
Figure 6. Comparison between actual (green solid line) and predicted trajectories of each object, with feedback (blue dashed line) and without feedback (red dotted line) for four different manipulation scenarios. Arrows show the corresponding object orientation. For better visualization the orientation arrow is only shown once for every 10 points in the trajectory. The actual trajectory is from data collected using motion capture. Feedback is every 2 seconds.

Table 1. Success rate (%) in simulation experiments using the proposed method.

| Task       | 1  | 2  | 3  | 4  |
|------------|----|----|----|----|
| Walker     | 96 | 66 | 70 | 56 |
| Blue Chair | 50 | 24 | 50 | 32 |
| Gray Chair | 60 | 24 | 38 | 48 |
| Rack       | 16 | 36 | 0  | 0  |

As we can see, position errors are generally lower than the orientation errors. This is because our robot can not apply torque directly and has to reach the desired orientation by only applying force. This, along with the limitations in robot motion, makes refining the final orientation very difficult.

In terms of success, the walker has the highest success rate. We believe the reason for this is that we have defined all weights and scenarios using the walker object and used the same weights and scenarios for all other objects without any modification. For example, our last object which is a rack is a large object and as we can see two of the scenarios were not suitable for that object at all due to the limited space. In addition, tuning optimization weights can affect the overall performance of the proposed method. With a more systematic weight tuning, we can improve the results for other objects.

Moreover, the object’s rotational inertia plays an important role in the final orientation success. A higher rotational inertia means that the object needs more torque for rotation which is harder to perform by only applying force to one of the object’s legs. We can see this effect in the second object which is a 4-wheeled heavy chair.

Running on a Core i7 2.4GH platform, the computational time for each step is less than a millisecond which is
Figure 7. Example of simulation results for the walker. Figures on right show the trajectories using LQR and on the left are resulting trajectories using the proposed method. The red object shows the goal configuration. In this figures, for better visualization, we only show the manipulation part of planning excluding motion planning, grasping and repositioning modes.
Figure 8. Final position and orientation errors for all objects through all tasks. The x-axis shows the task number. We assume that the maximum allowed run time in all trials is 3 minutes and the system stops after that even if it has not reach the goal region yet. The goal region is defined as distance less than 10cm to the goal position with less than 0.2 rad deviation from the goal orientation which is shown as red dashed line.

Results from two tasks, each with 5 trials are reported in Fig. 11 and Table 2. We can see that the second task (40% success rate) is more difficult than the first one (80% success rate). We believe this is mainly because of the longer distance in the second task which needs more repositioning actions which adds more error and opportunities to fail. A better re-grasp planning approach would improve the performance.
Figure 9. Frames of the physical experiment performing the first task. (1) light pink line: robot’s planned path, (2) light blue line: object’s planned path, (3) dark pink line: robot’s actual path, (4) dark blue line: object’s actual path. The desire object configuration is also shown as a simulated walker with the blue sphere showing its center.

Figure 10. Frames of the physical experiment performing the second task. (1) light pink line: robot’s planned path, (2) light blue line: object’s planned path, (3) dark pink line: robot’s actual path, (4) dark blue line: object’s actual path. The desire object configuration is also shown as a simulated walker with the blue sphere showing its center. In this task, the robot had to reposition once near the end of trajectory.
Conclusion

We presented an optimization-based framework for mobile manipulation. We focused on the problem of moving large legged objects in which we have to choose between legs to push or pull. We implemented a Bayesian regression method for autonomous learning of approximate dynamic parameters given 3 different models. We show that a simple “point mass on a wheel” model is sufficient for our application. However, it is possible to use more complicated models as well.

We use mixed-integer convex optimization to solve the hybrid control problem comprised of i) choosing which leg to grasp, and ii) continuous applied forces to move the object. Using MPC lets the system recover from modeling errors and find an optimal path to manipulate the object to a desired configuration. We validated our algorithm in simulated problems and real-world experiments. In simulations, we investigate the effect of replanning by comparing our algorithm with LQR.

In the process, we also found that the optimization weights have a significant effect on the performance of planning and a systematic method to assign those for each object should be found. As future work, we would like to conduct physical experiments with a more powerful mobile robot and use other objects with different wheel configurations. In addition, a better obstacle avoidance approach which is not as conservative as Big-M method would leave more space for the robot to maneuver, probably leading to a higher success rate.

Another interesting possibility for future research is grasp planning of legged objects considering the manipulation plan. In other words, planning the grasp position so that the robot grasps the leg from the best possible direction in order to increase the amount of manipulation with the same grasp. This will decrease the need for repositioning.

Finally, this planning approach should be validated with real perception. Moreover, we would like to perform user studies to evaluate the performance of the algorithm in delivering mobility aids to humans.

References

Arruda E, Mathew MJ, Kopicki M, Mistry M, Azad M and Wyatt JL (2017) Uncertainty averse pushing with model predictive path integral control. arXiv preprint arXiv:1710.04005.

Berenson D, Kuffner J and Choset H (2008) An optimization approach to planning for mobile manipulation. In: 2008 IEEE International Conference on Robotics and Automation. IEEE, pp. 1187–1192.

Bernardo JM and Smith AF (2001) Bayesian theory. Cambridge university press.

Casiddu N, Cesta A, Cortellessa G, Orlandini A, Porfirievo C, Divano A, Micheli E and Zallio M (2015) Robot interface design: The giraff telepresence robot for social interaction. In: Ambient Assisted Living. Springer, pp. 499–509.

Cehajic D, Dohmann PBG and Hirche S (2017) Estimating unknown object dynamics in human-robot manipulation tasks. Proceedings - IEEE International Conference on Robotics and Automation: 1730–1737DOI:10.1109/ICRA.2017.7989204.

Chen T, Ciocarlie M, Cousins S, Grice PM, Hawkins K, Hsiao K, Kemp C, King CH, Lazewatsky D, Leeper AE, Nguyen H, Paepcke A, Pantofaru C, Smart W and Takayama L (2013) Robots for Humanity: A Case Study in Assistive Mobile Manipulation. IEEE Robotics & Automation Magazine, Special issue on Assistive Robotics 20(1). URL http://www.willowgarage.com/sites/default/files/Chen___RFH___ram___12013.pdf.

Chu RZ (2017) Preventing in-patient falls: The nurse’s pivotal role. Nursing2017 47(3): 25–30. DOI:10.1097/01.NURSE.0000512872.83762.69.

Ciocarlie M, Hsiao K, Leeper A and Gosswos D (2012) Mobile manipulation through an assistive home robot. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 5313–5320.

Desai JP and Kumar V (1997) Nonholonomic motion planning for multiple mobile manipulators. In: IEEE International Conference on Robotics and Automation (ICRA), volume 4. IEEE, pp. 3409–3414.

Fitzpatrick P, Metta G, Natale L, Rao S and Sandini G (2003) Learning about objects through action-initial steps towards artificial cognition. In: IEEE International Conference on Robotics and Automation (ICRA), volume 3. IEEE, pp. 3140–3145.
Franchi A, Petitti A and Rizzo A (2015) Decentralized parameter estimation and observation for cooperative mobile manipulation of an unknown load using noisy measurements. Proceedings - IEEE International Conference on Robotics and Automation 2015-June(June): 5517–5522. DOI:10.1109/ ICRA.2015.7139970.

Gurobi Optimization Inc (2018) Gurobi optimizer reference manual. URL http://www.gurobi.com.

Hermans T, Li F, Rehg JM and Bobick AF (2013) Learning contact locations for pushing and orienting unknown objects. In: Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on. IEEE, pp. 435–442.

Hogan FR, Grau ER and Rodriguez A (2017) Reactive planar manipulation with convex hybrid mpc. arXiv preprint arXiv:1710.05724.

Hogan FR and Rodriguez A (2016) Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics. arXiv preprint arXiv:1611.08268.

Joint C (2015) Preventing falls and fall-related injuries in health care facilities. Sentinel Event Alert 55: 1–5. URL http://www.ncbi.nlm.nih.gov/pubmed/26422837.

Körding KP and Wolpert DM (2004) Bayesian integration in sensorimotor learning. Nature 427(6971): 244.

Kretzschmar H, Spies M, Sprunk C and Burgard W (2016) Socially compliant mobile robot navigation via inverse reinforcement learning. The International Journal of Robotics Research 35(11): 1289–1307.

Kristoffersson A, Coradeschi S and Loutfi A (2013) A review of mobile robotic telepresence. Advances in Human-Computer Interaction 2013: 3.

Li Q and Payande S (2007) Manipulation of convex objects via two-agent point-contact push. The international journal of robotics research 26(4): 377–403.

Marino A and Pierrri F (2018) A two stage approach for distributed cooperative manipulation of an unknown object without explicit communication and unknown number of robots. Robotics and Autonomous Systems 103: 122–133. DOI: 10.1016/j.robot.2018.02.007. URL https://doi.org/10.1016/j.robot.2018.02.007.

Mason MT (1986) Mechanics and planning of manipulator pushing operations. The International Journal of Robotics Research 5(3): 53–71.

Mast M, Burmester M, Graf B, Weishardt F, Arbeiter G, Španěl M, Materna Z, Smrž P and Kronreif G (2015) Design of the human-robot interaction for a semi-autonomous service robot to assist elderly people. In: Ambient Assisted Living. Springer, pp. 15–29.

Nakhaeinia D, Payeur P, Hong TS and Karasfi B (2015) A hybrid control architecture for autonomous mobile robot navigation in unknown dynamic environment. In: IEEE International Conference on Automation Science and Engineering (CASE). IEEE, pp. 1274–1281.

Ogata T, Ohba H, Tani J, Komatani K and Okuno HG (2005) Extracting multi-modal dynamics of objects using mnpb. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 966–971.

Oliver D, Britton M, Seed P, Martin F and Hopper A (1997) Development and evaluation of evidence based risk assessment tool (stratify) to predict which elderly inpatients will fall: case-control and cohort studies. Bmij 315(7115): 1049–1053.

Pajarinen J, Kyriki V, Koval M, Srinivasa S, Peters J and Neumann G (2017) Hybrid control trajectory optimization under uncertainty. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 5694–5701.

Patil A, Huard D and Fonnesbeck CJ (2010) Pymc: Bayesian stochastic modelling in python. Journal of statistical software 35(4): 1.

Pol RS and Murugan M (2015) A review on indoor human aware autonomous mobile robot navigation through a dynamic environment survey of different path planning algorithm and methods. In: International Conference On Industrial Instrumentation and Control (ICIC). IEEE, pp. 1339–1344.

Pripf J, Körtner T, Batko-Klein D, Hebesberger D, Weninger M, Gisinger C, Frennert S, Eftring H, Antona M, Adami I, Weiss A, Bajones M and Vincze M (2016) Results of a real world trial with a mobile social service robot for older adults. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction. IEEE Press, pp. 497–498.

Richards A and How J (2005) Mixed-integer programming for control. In: Proceedings of the American Control Conference. IEEE, pp. 2676–2683.

Sabbagh Novin R, Yazdani A, Hermans T and Merryweather A (2018) Dynamic model learning and manipulation planning for objects in hospitals using a patient assistant mobile (pam) robot. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 1–7.

Salganicoff M, Metta G, Oddera A and Sandini G (1993) A vision-based learning method for pushing manipulation, volume 54. University of Pennsylvania.

Scholz J, Jindal N, Levihn M, Isbell CL and Christensen HI (2016) Navigation among movable obstacles with learned dynamic constraints. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 3706–3713.

Scholz J, Levihn M, Isbell CL, Christensen H and Stilman M (2015) Learning non-holonomic object models for mobile manipulation. In: IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 5531–5536.

Stilman M, Nishiwaki K and Kagami S (2007) Learning object models for whole body manipulation. In: 7th IEEE-RAS International Conference on Humanoid Robots. IEEE, pp. 174–179.

Sun Y, Xi N, Tan J and Wang Y (2002) Interactive model identification for nonholonomic cart pushed by a mobile manipulator. In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), volume 4. IEEE, pp. 3966–3971.

Vithani AR and Gupta KC (2002) Estimation of object kinematics from point data. In: ASME 2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers, pp. 395–403.

Woodruff JZ and Lynch KM (2017) Planning and control for dynamic, nonprehensile, and hybrid manipulation tasks. In: IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 4066–4073.