Planning for Primitive Manipulations

Neel Doshi and Alberto Rodriguez
Dept. of Mechanical Engineering – Massachusetts Institute of Technology
<nddoshi, albertor>@mit.edu

***Abstract*—We present a model-based approach for planning primitive manipulations with pre-defined environmental contact(s). Given the environmental contact(s) and a set of available robot contacts, our framework finds contact-location and contact-force trajectories that drive the object from its current pose to the goal. We efficiently solve the underlying mixed-integer non-linear program (MI-NLP) by developing a model that separates the mixed-integer and non-linear portions of the mechanics. We then design an iterative-lqr (iLQR) based algorithm that exploits this structure to find the contact-location trajectory during the backward pass and then solve for the contact-forces and corresponding object poses in the forward pass. We apply our approach to several well-known manipulation primitives, including grasping, pushing, pulling, and pivoting, and find that our algorithm can efficiently (in 1 s to 6 s) plan pose-to-pose object manipulations.**

I. Introduction

The ability to autonomously manipulate objects will be critical to successful on-orbit robotic assembly and satellite repair. Here we focus on planning object manipulations by sequencing manipulation primitives. Complex manipulations, such as adding a component to an existing structure during an assembly task, can often be decomposed into a sequence of simpler (primitive) behaviors. Fig. 1 shows one possible primitive sequence for an assembly task: pulling the object along the structure's surface, pivoting the object to align it correctly, and then pushing it into the desired location.

One key feature of the three primitives described above is the existence of a fixed or pre-defined environmental contact. Indeed, many well-known manipulation primitives, including grasping [1], pulling [2], pushing [3], and pivoting [4] embody this idea. Based on this observation, we develop a model to describe the mechanics of robots manipulating objects with pre-defined environmental contacts. Our model allows for both robot and environmental contacts to have point, line, or patch geometry, and the environmental contacts can either stick or slide. More importantly, we identify that our model is non-linear, but continuous, in the object's pose and is hybrid, but affine, in contact forces. This enables the design of an iterative-lqr (iLQR) based algorithm that exploits this structure for efficient planning.

II. Manipulation with pre-defined contacts

We first present a model for manipulations such as those shown in Fig. 1. Our model is based on the following (fairly standard) implicit assumptions: known (polygonal) geometry of the object and contacts, known coefficients of friction, rigid-body interaction, quasi-static interaction, Coulomb's frictional laws, and full object state feedback. In addition

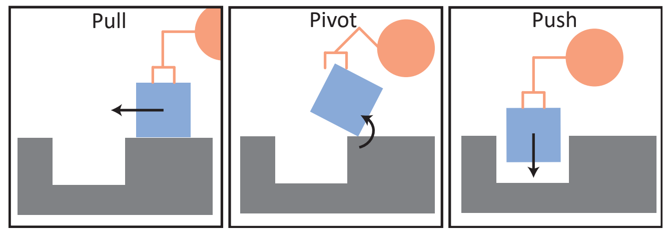

Fig. 1. Inserting a part into an existing structure by sequencing three manipulation primitives with fixed environmental contacts: a pull, a pivot, and a push.

we ask the user to pre-define environmental contacts and a set of available robot contacts. For example, in the Sagittal pivoting primitive (Fig. 1, middle panel), the user specifies that the object makes a point contact with the ground on one corner and available robot contacts could be located at the other three corners of the object. Note, that the user could have just as easily chosen the available robot contacts as lines on the other three sides of the object. Given these assumptions, a quasi-static model for this manipulation consists of the following four components: a forward-model (1), static equilibrium constraints (2), friction-cone constraints (3), and kinematic constraints imposed by the environmental contacts (4). More formally, the model is described by following set of hybrid differential algebraic equations:

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \Delta t \mathbf{R}(\mathbf{q}_i)\mathbf{v}_i, \tag{1}$$

$$\left(\mathbf{G}_{\text{ext}}\mathbf{w}_{\text{ext}} = \sum_{k=1}^{p} \mathbf{w}_k\right)_m, \tag{2}$$

$$\left(\mathbf{C}_k(\mathbf{q}_i)\mathbf{w}_k \geq \mathbf{d}_k(\mathbf{q}_i), \qquad k = 1,\ldots,p,\right)_m \tag{3}$$

$$\left(\mathbf{E}_k(\mathbf{q}_i)\mathbf{v}_i \geq \mathbf{f}_k(\mathbf{q}_i), \qquad k = 1,\ldots,p\right)_m. \tag{4}$$

Here $\mathbf{q}$ is the object's pose, $\mathbf{v}$ is the object's velocity, $\mathbf{R}(\mathbf{q}_i)$ is the rotation from body to world frame, Δt is the time-step, $\mathbf{w}_{\text{ext}}$ is the external (gravitational) wrench, and $\mathbf{w_k}$ are the wrenches applied at the k-th contact (robot or environmental). Note that while (1) is independent of the of the mode (m), equations (2)-(4) vary with m. The mode defines both the set of active contacts and their state (e.g., sticking or siding). In the Sagittal pivoting example from Fig. 1, the mode $j = 1$ might correspond to "top and right robot contacts active and sticking and environmental contact sliding left. This would then effect the active contacts and the kinematic constraints imposed on the object's velocity.

The linearity with respect to the contact wrench in (3)

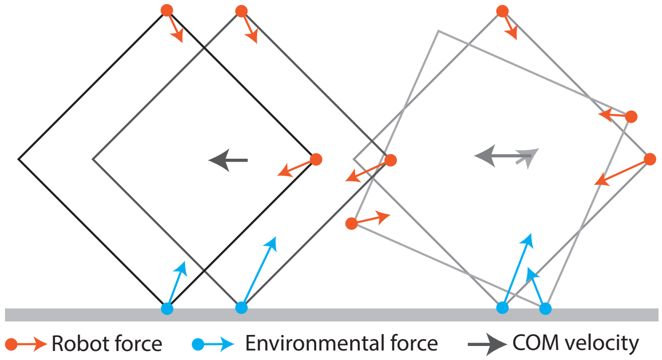

Fig. 2. Representative object-state, contact-location, and contact-force trajectories for the Sagittal pushing primitive. The object moves from the initial condition (lightest gray on the right) to the goal (black on the left).

is achieved by taking a polygonal approximation of the friction cone [5] and approximating line and patch contacts with collections of points [6]. Moreover, the linearity with respect to the object-velocity in (4) is achieved by treating each sliding boundary of the polygonal wrench-cone as a separate mode. Finally, note that the forward model (1) is continuous in both the state ($\mathbf{q}$) and input ($\mathbf{v}$, Δt, and $\mathbf{w}$) and the constraints (2) are affine in the input.

III. Hybrid iLQR

Our planner extends the idea of iterative LQR (iLQR) with *strong variations* [7], [8] (i.e., large variations in input) to the types of systems described in Section II. We rewrite the model developed above as

$$\mathbf{x}_{i+1} = \mathbf{f_1}(\mathbf{x}_i) + \mathbf{F_2}(\mathbf{x}_i)\mathbf{u}_i, \tag{5}$$

$$\Big(\mathbf{A}_k(\mathbf{x}_i)\mathbf{u}_i \geq \mathbf{b}_k(\mathbf{x}_i), \qquad k = 1,\dots,p\Big)_m, \tag{6}$$

where (5) is the forward model and (6) represents the constraints (2)-(4). Note here that $\mathbf{x} = \mathbf{q}$ and $\mathbf{u}$ is the concatenation of $\mathbf{v}$, Δt, and $\mathbf{w}_k$. Given a one-step cost $l(\mathbf{x}_i, \mathbf{u}_i)$ and final cost $l_f(\mathbf{x}_N)$, the optimal cost from a given state can be defined recursively via Bellman's equation

$$V^i(\mathbf{x}_i) = \min_{\mathbf{u}_i, m_i} \left[l(\mathbf{x}_i, \mathbf{u}_i) + V^{i+1}(\mathbf{q_{i+1}}) \right], \tag{7}$$

with the boundary condition $V^N = l_f(\mathbf{x}_N)$. Note here we are minimizing over both the continuous input $\mathbf{u_i}$ and the mode m_i; however, we only need to make minor modifications to the standard iLQR algorithm as (5) is smooth in $\mathbf{u}$. In particular, we use iLQR with *strong variations* and only expand (7) about a nominal state. This ability to apply larger variations in input allows us to search across different modes during the backwards pass. Moreover, by leveraging the control affine structure of both (5) and (6), we can still efficiently solve for a cost-improving mode and input.

Here we present an expansion of (7) about a nominal state ($\mathbf{\bar{x}_i}$) derived by taking a quadratic expansion of l and substituting a linear approximation of the forward dynamics (5) for $\mathbf{x}_{i+1}$:

$$v^i(\mathbf{\bar{x}_i} + \delta\mathbf{x_i}) = \min_{\mathbf{u_i}, j} \frac{1}{2} \begin{bmatrix} 1 \\ \delta\mathbf{x_i} \\ \mathbf{u_i} \end{bmatrix}^T \begin{bmatrix} 2q_0^i & (\mathbf{q_x^i})^\mathbf{T} & (\mathbf{q_u^i})^\mathbf{T} \\ \mathbf{q_x^i} & \mathbf{Q_{xx}^i} & \mathbf{Q_{xu}^i} \\ \mathbf{q_u^i} & \mathbf{Q_{ux}^i} & \mathbf{Q_{uu}^i} \end{bmatrix} \begin{bmatrix} 1 \\ \delta\mathbf{x_i} \\ \mathbf{u_i} \end{bmatrix}. \tag{8}$$

Note that the expression for expansion coefficients are omitted for brevity. We can now perform the minimization in (8) subject to the constraints in (6). This amounts to solving a small mixed-integer quadratic program (MIQP) to find both the mode (m_i) and continuous inputs ($\mathbf{u}_i$) that minimize the local approximation of (7). Substituting this back into (8) and performing some algebra, we get:

$$V^i = q_0^i + (\mathbf{q_u^i})^T \mathbf{u_0^i} + \frac{1}{2}(\mathbf{u_0^i})^T \mathbf{Q_{uu}^i} \mathbf{u_0^i} \tag{9}$$

$$\mathbf{v_x^i} = \mathbf{q_x^i} + \mathbf{Q_{xu}^i}(\mathbf{Q_{uu}^i})^{-1}\mathbf{q_u^i} \tag{10}$$

$$\mathbf{V_{xx}^i} = \mathbf{Q_{ux}^i} - \frac{1}{2}\mathbf{Q_{xu}^i}(\mathbf{Q_{uu}^i})^{-1}\mathbf{Q_{ux}^i} \tag{11}$$

We can now start at the terminal step $i = N$, with $V^N = l(\mathbf{x}_N)$. Integrating (9)-(11) constitutes a backwards pass of our approach, and the forward pass can then be conducted as follows:

$$\mathbf{\hat{x}_0} = \mathbf{\bar{x}_0} \tag{12}$$

$$\mathbf{\hat{u}_i} = \mathbf{u_i} + \mathbf{K^i}(\mathbf{\hat{x}_i} - \mathbf{\bar{x}_i}) \qquad i = 0,\dots,N-1 \tag{13}$$

$$\mathbf{\hat{x}_{i+1}} = \mathbf{f^1}(\mathbf{\hat{x}_i}) + \mathbf{f^2}(\mathbf{\hat{x}_i})\mathbf{\hat{u}_i} \qquad i = 0,\dots,N-1 \tag{14}$$

where $\mathbf{K}^i$ is defined as in [9], $\mathbf{\hat{x}_i}$ and $\mathbf{\hat{u}_i}$ form the trajectory for the next iteration of the algorithm. Note that details related to line-searching and regularization are omitted for brevity.

IV. Simulated Examples

Here we use our algorithm to plan trajectories for the primitives shown in Fig. 1. A representative trajectory for the Sagittal pivoting primitive is shown in Fig. 2. We plan over a $N = 20$ step horizon, and at each step the planner chooses the location of two active robot contacts and the state of the environmental contact (nine hybrid modes). The algorithm both finds the contact locations and forces that drive the object to the goal in ∼6.5 s. We are also able to efficiently plan 20 step trajectories for the other three primitives from Fig. 1: Sagittal pushing (six modes), horizontal pushing (four modes), and horizontal pulling (six modes). The planning times for those trajectories are ∼3 s, ∼1.75 s, and ∼2.25 s, respectively. In our current implementation, the computational complexity per iteration is dominated by the MIQPs and scales with pN, where p is the number of modes and N is the planning horizon.

V. Conclusions

In summary, we introduce an approach for planning finite-horizon object-state, contact-location, and contact-force trajectories for manipulations with fixed environmental contacts. Our approach consists of developing a quasi-static model of these manipulation that exposes underlying structure which is then leveraged by our planner, a hybrid extension to iLQR with strong variations.

References

[1] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer handbook of robotics*. Springer, 2016, pp. 955–988.

[2] M. M. Ghazaei Ardakani, J. Bimbo, and D. Prattichizzo, "Quasi-static analysis of planar sliding using friction patches," *The International Journal of Robotics Research*, p. 0278364920929082, 2020.

[3] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[4] Y. Hou, Z. Jia, and M. T. Mason, "Fast planning for 3d any-pose-reorienting using pivoting," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1631–1638.

[5] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction," *International Journal for Numerical Methods in Engineering*, vol. 39, no. 15, pp. 2673–2691, 1996.

[6] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6215–6222.

[7] D. H. Jacobson and D. Q. Mayne, *Differential dynamic programming*. North-Holland, 1970.

[8] S. B. GERSHWIN and D. H. JACOBSON, "A discrete-time differential dynamic programming algorithm with application to optimal orbit transfer," *AIAA Journal*, vol. 8, no. 9, pp. 1616–1626, 1970.

[9] D. M. Murray and S. J. Yakowitz, "Constrained differential dynamic programming and its application to multireservoir control," *Water Resources Research*, vol. 15, no. 5, pp. 1017–1027, 1979.