Convex Quasi-Dynamic Simulation of Rigid Point Clouds with Torsional Friction

Kevin Tracy[1] and Zachary Manchester[1]

***Abstract*— Many common manipulation tasks move slowly enough that velocities and Coriolis forces do not contribute meaningfully to the dynamics of the robot. As a result, quasi-dynamic simulation of these systems, in which all forces are assumed to be in equilibrium, can produce physically accurate results for use in motion-planning and control. Recent work has combined the quasi-dynamic model with a relaxed formulation of Coulomb friction, where the resulting dynamics are the solution to a convex quadratic program. We extend this recent work by directly manipulating rigid point clouds, without the need for meshing or decomposition into convex primitives. We also introduce a novel torsional friction model to mimic the frictional behaviors of the sorts of patch contacts that exist on real systems. These ideas are demonstrated in a grasping example, where a dense point cloud is manipulated with and without torsional friction, clearly showing the utility of the torsional friction model.**

I. Introduction

Accurate and computationally efficient simulation for contact-rich robotic manipulation tasks is crucial for developing model-based control policies. Existing simulators like Bullet [1], Drake [2], Dart [3], and MuJoCo [4], are able to model complex contact and friction interactions between a wide variety of geometries. While these simulators evaluate the full second-order dynamics of a robot, for many manipulation tasks, these second-order dynamics are not always necessary, and substantial computational savings are possible.

For manipulation systems where objects are generally moving slowly, the velocities of the bodies are small enough that accelerations and Coriolis forces do not impact the dynamics in a meaningful way. As a result, quasi-dynamic [5] simulation methods, in which all forces are assumed to be in equilibrium, can be employed to solve for displacements in the configurations at each time step.

This work is inspired by [6], where a relaxed friction model from [7] is incorporated into a quasi-dynamic simulation method that treats actuators as impedances. Each simulation step consists of solving a small, well-defined convex Quadratic Program (QP), making the method fast and efficient, with virtually none of the stability problems encountered with full second-order dynamics formulations. This quasi-dynamic model was used successfully in a global planner in [8] where it was leveraged to build sophisticated motion plans for common manipulation tasks. In both of these works, only relatively simple shapes like spheres, capsules, and boxes were considered, due to the complications involved in collision checking.

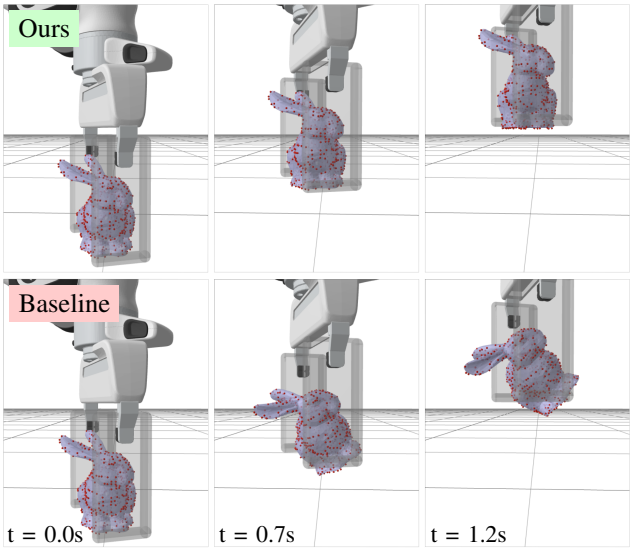

Fig. 1. Quasi-dynamic simulation of grasping a point cloud with and without torsional friction on the points. In the bottom sequence of frames, a simulation without torsional friction results in non-physical rotation of the bunny about single-point contacts with each gripper. Our method (top) includes torsional Coulomb friction at each contact point to avoid this behavior.

In this paper, we make two contributions to the state of the art for quasi-dynamic simulation: the first is a computationally efficient framework for computing the dynamics of rigid objects whose complex geometry is represented by point clouds, and the second is a convex torsional friction model to effectively model patch contacts and avoid some of the non-physical behaviors seen when manipulating a rigid point cloud with rigid manipulators. Since the new torsional friction model is also represented as a convex constraint, each simulation step can still be computed efficiently by solving a QP. Together, these contributions enable fast and robust simulation of manipulation tasks involving complex non-convex geometries that would otherwise be challenging to decompose into simpler primitives.

II. Convex Quasi-Dynamic Models

This section describes the general quasi-dynamic formulation with actuators modeled by impedances, closely following [6], [9]. The configuration of the whole system, $q \in \mathbf{R}^{n_u+n_a}$, can be partitioned into an actuated part, $q_a \in \mathbf{R}^{n_a}$, and an un-actuated part $q_u \in \mathbf{R}^{n_u}$, stacked

[1]Kevin Tracy and Zachary Manchester are with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA {ktracy,zacm}@cmu.edu

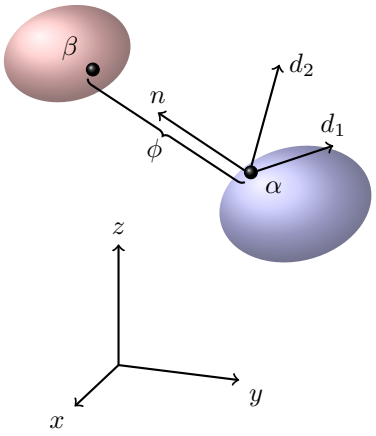

Fig. 2. Description of contact geometry for a given signed distance function $\phi = \|\beta - \alpha\|$, where α and β are the closest points between two convex shapes. The unit contact normal vector n extends from α to β, with a set of orthogonal tangent-plane vectors d_1 and d_2.

as $q = [q_a^T, q_u^T]^T$. In a quasi-dynamic model, there is no accumulation of velocity between time steps. As a result, a new velocity or a configuration displacement is computed at each time step.

The actuated degrees of freedom are treated as if they were controlled with impedance control. This means that we have some diagonal joint stiffness matrix $K_q \in \mathbb{S}^{n_a \times n_a}$, with the generalized force τ_k modeled as:

$$K_q \delta \bar{q}_a = \tau_k, \tag{1}$$

where $\delta \bar{q}_a$ is the desired change in actuated configuration. The un-actuated degrees of freedom are simply acted upon by an external generalized force τ_a, and contact forces.

A. Contact and Friction Model

Rigid contact and Coulomb friction can be incorporated into a quasi-dynamic model in the same fashion as a second-order dynamic model. First, collision information between pairs of convex objects can be computed with any number of methods, given that they return the closest points between objects as well as a contact normal vector [10]–[13]. As shown in figure 2, these closest points are denoted α and β, with the signed-distance function (SDF) between these denoted by ϕ. Each of the n_c pairwise contact interactions has its own α, β, normal vector pointing from α to β, and a set of orthogonal tangent vectors d_1 and d_2. Together, the set $\{d_1, d_2, n\}$ describe a dextral triad of orthonormal vectors [14], [15].

In [7], Anitescu introduces a convex relaxation of tangential Coulomb friction that is exact when objects are sticking, and only introduces a slight "boundary layer" when objects are sliding. This relaxation allows for the inclusion of friction as a linear inequality in a quadratic program, enabling convex time-stepping methods [9]. In this work, we extend Anitescu's friction model to handle torsional friction about the contact normal. This section omits indices that specify which contact pair is being considered for clarity, but the method extends to an arbitrary number of contacts.

A Jacobian mapping forces $\lambda_i \in \mathbf{R}^3$ at the contact point into generalized coordinates is calculated by taking the Jacobian of the following function with respect to the generalized velocity $v \in \mathbf{R}^{n_v}$:

$$f(q, v) = \begin{bmatrix} \nu_\beta - \nu_\alpha \\ n^T(\omega_\beta - \omega_\alpha) \end{bmatrix}, \tag{2}$$

where the relative velocity of the contact points is calculated by taking the difference between the velocity of β ($\nu_\beta \in \mathbf{R}^3$) and α ($\nu_\alpha \in \mathbf{R}^3$). This Jacobian will be referred to a J:

$$J = \frac{\partial f(q, v)}{\partial v}. \tag{3}$$

where each vector in (2) is resolved in the world frame. With this Jacobian, we can now write our friction constraints for a given point as the following:

$$\phi + [J^T(\tilde{n} + \tilde{d}_i)]^T \delta q \geq 0 \quad \forall i \tag{4}$$

where $\tilde{n} = [n^T, 0]^T$, and the the first four $\tilde{d}$'s describing the standard pyramidal tangential friction-cone approximation are the following:

$$\tilde{d}_1 = \begin{bmatrix} \mu_v d_1 \\ 0 \end{bmatrix}, \qquad \tilde{d}_2 = \begin{bmatrix} \mu_v d_2 \\ 0 \end{bmatrix}, \tag{5}$$

$$\tilde{d}_3 = \begin{bmatrix} -\mu_v d_1 \\ 0 \end{bmatrix}, \qquad \tilde{d}_4 = \begin{bmatrix} -\mu_v d_2 \\ 0 \end{bmatrix}. \tag{6}$$

We also introduce two more $\tilde{d}$'s describing torsional friction:

$$\tilde{d}_5 = \begin{bmatrix} 0_3 \\ \mu_\omega \end{bmatrix}, \qquad \tilde{d}_6 = \begin{bmatrix} 0_3 \\ -\mu_\omega \end{bmatrix}, \tag{7}$$

where $\mu_v \in \mathbf{R}_+$ is the tangential coefficient of friction and $\mu_\omega \in \mathbf{R}_+$ is the torsional coefficient of friction. These six linear inequalities describe the contact and friction for a given pair and can be vertically concatenated into a single $A\delta q \geq b$ constraint for convenience:

$$A = \begin{bmatrix} J^T(\tilde{n} + \tilde{d}_1) \\ J^T(\tilde{n} + \tilde{d}_1) \\ \vdots \\ J^T(\tilde{n} + \tilde{d}_6) \end{bmatrix}, \qquad b = -\phi 1_6. \tag{8}$$

B. Optimization Formulation

The force balance for the actuated and un-actuated components of the configuration with contact forces can be posed as,

$$\begin{bmatrix} hK_q \delta q_a - hK_q \delta \bar{q}_a + \tau_a \\ \frac{1}{h} M_u - h\tau_u \end{bmatrix} + \sum_n^{n_c} A_j^T \lambda_j = 0, \tag{9}$$

where the contact/friction model is described as follows:

$$A_j \delta q \geq b_j, \tag{10}$$

$$\lambda_j \geq 0, \tag{11}$$

$$(A_j \delta q - b_j) \circ \lambda = 0. \tag{12}$$

These conditions are the KKT optimality conditions for a convex Quadratic Program (QP) where (9) is stationarity,

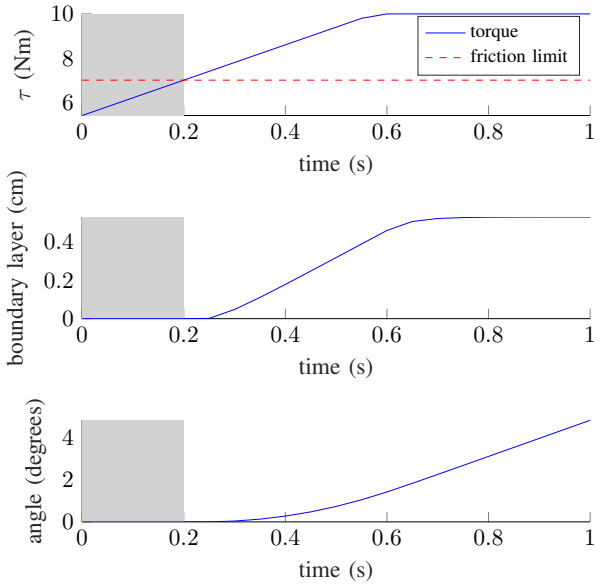

Fig. 3. Visualization of the boundary layer that exists only when a point in contact begins rotating due to an external torque τ. In the exact same fashion as the tangential friction constraints, a boundary layer is introduced during sliding that is proportional to the time step h.

(10) is primal feasibility, (11) is dual feasibility, and (12) is complementarity [16]. This QP is as follows:

$$\begin{aligned} \min_{\delta q} \quad & \frac{1}{2}\delta q^T \begin{bmatrix} hK_q & 0 \\ 0 & M_u/h \end{bmatrix} \delta q - h \begin{bmatrix} K_q \delta \bar{q}_a + \tau_a \\ \tau_u \end{bmatrix}^T \delta q \\ \text{s.t.} \quad & A_j \delta q \geq b_j, \qquad \forall j \in \mathcal{E} \end{aligned} \tag{13}$$

where $\mathcal{E}$ is the set of contact pairs being considered and $h \in \mathbf{R}_+$ is the time-step size. For large point clouds, this pruning of constraints is critical for keeping the resulting QP small. Another benefit to the QP formulation is differentiability: In recent years, differentiable convex optimization has enabled smooth automatic differentiation through QP solvers with good results [17]–[20].

III. Numerical Examples

Two examples are shown to demonstrate the behavior and utility of our formulation: The first example is a simple spinning sphere that showcases the tightness of the relaxed friction constraints, and the second is a more practical example of gripping a dense point cloud with and without torsional friction.

A. Torsional Boundary Layer

To demonstrate the boundary layer induced by the relaxed torsional friction constraints, figure 3 demonstrates what happens when a sphere in contact with the floor is spun. An increasing external torque is applied to the sphere, and up until the maximum allowable friction torque is saturated, the sphere doesn't move. Once the external torque exceeds this limit, the sphere begins to spin and lifts off from the surface by a fraction of a centimeter.

This boundary-layer behavior mirrors the tangential friction case, where this is the only deviation from Coulomb friction and the size of this boundary layer decreases with step size. For most manipulation tasks that aren't pushing or sliding, full sticking behavior is desirable, making this relaxed model appropriate.

B. Grasping a Point Cloud

In this example, shown in Fig. 1, parallel grippers are used to grasp a point-cloud bunny with and without torsional friction. The bunny was modeled as a rigid body with 992 points, each of which was constrained to consider the floor, the left gripper, and the right gripper. The simulator checks the signed distance function for each of these 992 points with respect to the three objects, and selects five points for each object to include in the quadratic program. Each of the selected points contributes the constraints shown in (8) to the optimization.

While these five closest points in question can change from time step to time step, the size of the QP remains the same. The resulting QP for this example has 13 primal variables with 80 inequality constraints, and is solved with a custom primal-dual interior-point algorithm [21]–[23] that leverages templating based on the number of contacts considered during a simulation.

When the grippers close on the bunny, it is possible for there to be single points of contact with each gripper. In this case, the lack of torsional friction results in the bunny rotating about these contact points. While this is technically correct behavior given point contacts, the true system does not exhibit this behavior due to patch-contact effects, resulting in a significant sim-to-real gap. On a real robotic system, there is always some compliance either on the gripper or the object that manifests in a patch contact instead of a point contact. The patch contact provides a lever arm for torsional friction, preventing the bunny from pivoting. Just like with tangential friction, the torsional friction is proportional to the normal force on the contact.

By including torsional friction for each point contact, we are able to replicate torsional friction behaviors while still treating the object as a rigid point cloud.

The full simulation step takes less than 90 μs for a time step size of $h = 0.01s$., with the profile of this function shown in Fig. 4.[1] Since a majority of the time is spent solving the QP, there is reason to believe this sort of simulation makes more sense on a CPU than a GPU, despite the possibility of a very dense point cloud. There may be cases where the point cloud is dense enough that computing and sorting all of the pairwise contacts is the limiting computational factor, in which case a GPU implementation would make more sense.

References

[1] E. Coumans, "Bullet Physics Simulation," in *SIGGRAPH*, Los Angeles: ACM, 2015.

[1]Timing results are reflective of a Mac M1 Pro, running Julia 1.9.2

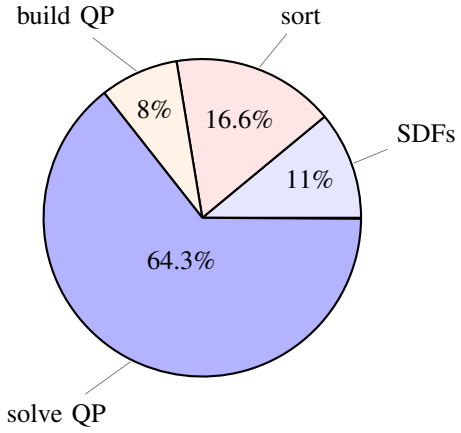

Fig. 4. Average timing results for a step in the quasi-dynamic simulator with the bunny where the total time was $<$90 μs for a time step size of $h = 0.01s$. A majority of the time is spent solving the QP, despite the fact there are almost 3,000 signed distance functions evaluations.

[2] R. Tedrake and The Drake Development Team, "Drake: Model-based design and verification for robotics," 2019.

[3] J. Lee, M. X. Grey, S. Ha, *et al.*, "DART: Dynamic Animation and Robotics Toolkit," *The Journal of Open Source Software*, vol. 3, no. 22, p. 500, Feb. 2018.

[4] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[5] M. T. Mason, *Mechanics of Robotic Manipulation*. The MIT Press, Jun. 2001.

[6] T. Pang and R. Tedrake, "A Robust Time-Stepping Scheme for Quasistatic Rigid Multibody Systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid: IEEE, Oct. 2018, pp. 5640–5647.

[7] M. Anitescu, "Optimization-based simulation of nonsmooth rigid multibody dynamics," *Mathematical Programming*, vol. 105, no. 1, pp. 113–143, Jan. 2006.

[8] T. Pang, H. J. T. Suh, L. Yang, and R. Tedrake, *Global Planning for Contact-Rich Manipulation via Local Smoothing of Quasi-dynamic Contact Models*, Feb. 2023.

[9] T. Pang and R. Tedrake, "A Convex Quasistatic Time-stepping Scheme for Rigid Multibody Systems with Contact and Friction," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China: IEEE, May 2021, pp. 6614–6620.

[10] E. Gilbert, D. Johnson, and S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, Apr. 1988.

[11] G. Snethen, "XenoCollide: Complex Collision Made Simple," *undefined*, 2008.

[12] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *2012 IEEE International Conference on Robotics and Automation*, St Paul, MN, USA: IEEE, May 2012, pp. 3859–3866.

[13] K. Tracy, T. A. Howell, and Z. Manchester, *Differentiable Collision Detection for a Set of Convex Primitives*, Jul. 2022.

[14] P. Mitiguy, *Advanced Dynamics & Motion Simulation*. Motion Genesis, Aug. 2018.

[15] T. R. Kane, P. W. Likins, and D. A. Levinson, *Spacecraft Dynamics*. New York: McGraw-Hill Book Co, 1983.

[16] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[17] K. Tracy, Z. Manchester, and E. Douglas, "Ultra-Fine Pointing for Nanosatellite Telescopes With Actuated Booms," in *2022 IEEE Aerospace Conference (AERO)*, Big Sky, MT, USA: IEEE, Mar. 2022, pp. 1–8.

[18] T. A. Howell, S. L. Cleac'h, J. Z. Kolter, M. Schwager, and Z. Manchester, "Dojo: A Differentiable Physics Engine for Robotics," *IEEE Transactions on Robotics and Automation (In Review)*, Jun. 2022.

[19] B. Amos and J. Z. Kolter, "OptNet: Differentiable Optimization as a Layer in Neural Networks," *arXiv:1703.00443 [cs, math, stat]*, Oct. 2019.

[20] H. J. Suh and R. Tedrake, "The Surprising Effectiveness of Linear Models for Visual Foresight in Object Pile Manipulation," *Springer Proceedings in Advanced Robotics*, vol. 17, pp. 347–363, Feb. 2020.

[21] S. Mehrotra, "On the Implementation of a Primal-Dual Interior Point Method," *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 575–601, Nov. 1992.

[22] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd. Springer, 2006.

[23] L. Vandenberghe, "The CVXOPT linear and quadratic cone program solvers," p. 30,