Robust and Safe Trajectory Planning for an Extendable Boom Manipulator

Stanley Wang and Steven Salah-Eddine

{swang11, stevense}@stanford.edu

June 6, 2024

Abstract

This paper presents a novel application of optimal control theory for the robust and safe trajectory planning of an extendable boom manipulator, a key component of ReachBot—a new robot platform designed for extreme environment exploration and manipulation. Using Sequential Convex Programming (SCP), we generate optimal end-effector trajectories between initial and goal positions. We derive and linearize the nonlinear state dynamics for a 3-degree-of-freedom RRP kinematic chain, incorporating state constraints, actuator limits, and a trust region around the linearization. Our findings show that actuator limits significantly influence trajectory shaping. Compared to PD control, the optimization-based approach demonstrates clear advantages. Finally, we validate our planned trajectories on physical hardware with a rock-grasping task.

1 *Introduction*

This project relates to ReachBot–a novel robot platform being developed by BDML and ASL at Stanford.

Current space ventures on the Moon and Mars have primarily relied on autonomous rovers for surface-level exploration. However, the critical task of robustly surveying more extreme environments–such as Martian lava tubes (Fig. 1) or lunar mare pits–is simply impossible with current solutions. Addressing this technological gap, ReachBot is a new robot design consisting of a small body with exceptional reach. We enable ReachBot to perform forceful interactions in rocky terrain by instrumenting lightweight deployable space booms with spined grippers. Combining several booms into a centralized body allows us to create an "arachnid-like" robot–capable of reconfiguring anchor points in the field for locomotion and manipulation.

Present work in BDML and ASL at Stanford has focused on the hardware and planning algorithms necessary for a complete realization of the ReachBot prototype. However, this goal is significantly limited by critical research gaps in the design and implementation of deployable boom manipulators, particularly concerning their feasibility and functionality in a complex robot system. While preliminary boom hardware systems constructed from deployable tape measures have shown promise, their performance has been limited by the application of simple control schemes (i.e. PID control). For this project, we propose a novel application of the optimal control framework presented in this course for **robust and safe trajectory planning of a boom manipulator**.

We find an optimal control framework to be necessary and advantageous in comparison to simpler classical techniques due to the unique mechanics of the extensible boom manipulator. For one, the boom manipulator's physics change substantially as it is extended. For example, larger accelerations are permissible when the boom is short (low inertia) but should certainly be avoided when the boom is extended (high inertia) to avoid catastrophic buckling. Similarly, the positional accuracy of the boom is affected by its extension length. Under similar loads, a long boom will experience more total deflection at the end-effector. Thus, to generate manipulator trajectories maximizing performance (positional accuracy and speed) while respecting the physical limits of the boom (buckling), we look towards optimal control theory for a solution.

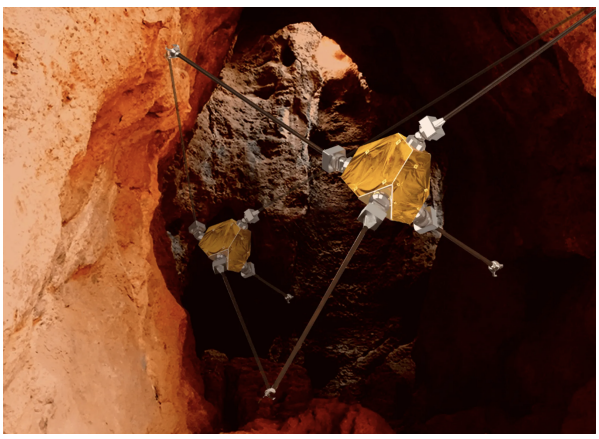

Fig. 1: Artistic rendition of ReachBot in a Martian cave (**Credits: Marco Pavone**)

2 Related Work

This work at present is a continuation of the ReachBot robot concept originally proposed by Chen et al. [2]. While this paper introduces the novel concept of using boom-mounted grippers for interfacing with extreme terrains, there are still extensive open questions regarding the feasibility and implementation of a highly dexterous boom manipulator for accomplish-

ing such tasks. The utility of an optimal control framework for ReachBot has been demonstrated by Morton et al. [5] for planning optimal full-robot stances with multiple booms/appendages. However, we seek to employ optimal control methods on a lower level (in improving the capabilities of a single boom) as a precursor to a full robot system.

There are several useful references to aid in the formulation of a dynamics model for our boom. Zhongyi et al. [3] provides an excellent method of applying static beam theory in calculating the bending and torsional stresses of a lenticular composite boom of specified geometry. Their work further extends to dynamic analysis with a strain energy formulation, followed by a sequential quadratic programming optimization of the boom design variables (balancing accuracy and efficiency). While trajectory planning is not considered, the formal treatment of boom dynamics is highly relevant to our application.

A more foundational work by Seffen et al. [6] considers the accuracy of deployment of a tape-spring beam with respect to bending stiffness. While trajectory optimization is not formally utilized in this work, their consideration of a compliant boom provides a good reference in devising our model.

Another study is by Cannon and Schmitz [1] on the endpoint control of a flexible one-link robot. This paper addresses the challenge of controlling flexible structures with non-colocated sensors and actuators, similar to ReachBot's boom manipulator system. Cannon and Schmitz demonstrated that non-colocated control could achieve stability and responsiveness with accurate dynamic models and advanced algorithms. Their experiments with a single-link flexible manipulator showed that achieving stability and high bandwidth required reduced-order and adaptive control methods. These findings are applicable in emphasizing the need for robust optimal control frameworks to manage dynamic behavior for ReachBot's boom manipulator.

Further groundwork for this research lies in the seminal study by Mason [4], which introduces a compliance-based approach for robotic manipulation. This work highlights the importance of integrating compliance in the design of robotic systems to adapt to unstructured environments, a principle directly applicable to ReachBot's deployment in extreme terrains. By devising compliance control strategies, ReachBot can achieve better adaptability and stability when interacting with irregular surfaces, thus enhancing its overall performance in Martian lava tubes and lunar mare pits.

Theodore and Ghosal [7] present an examination of flexible-link manipulators with prismatic joints, using the Euler-Bernoulli beam equation. This study is relevant to ReachBot as it tackles the complex nature of modeling and controlling a flexible boom manipulator for accurate trajectory planning. The authors' method of addressing moving boundary conditions and their solution to the time-dependent frequency equation provide a framework for understanding the dynamic behavior of flexible booms. These findings can be beneficial to enhance the modeling accuracy and control strategies for ReachBot's deployable boom.

3 Model Formulation

All models are wrong, but some are useful

- George E.P. Box

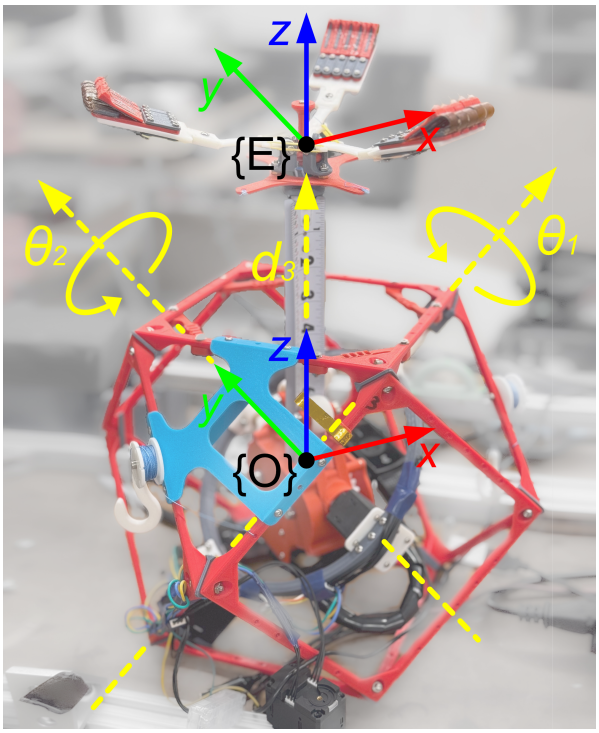

Fig. 2: Kinematic configuration of ReachBot's 3 DoF boom manipulator

3.1 Manipulator Kinematics

To formulate an appropriate model for the ReachBot boom manipulator, we begin by considering the fundamental kinematics of the manipulator. This allows us to establish an understanding of the relationship between the individually actuated joints and the position of the boom-mounted gripper (end-effector). Consider the schematic of the boom manipulator's kinematic structure as shown in Fig. 2

We classify the manipulator as an RRP architecture, short for "revolute-revolute-prismatic". Considering the kinematic chain from the robot's base to end-effector, we see a sequence of two gimbaled revolute joints θ_1, θ_2 followed by the boom's prismatic/translation joint d_3. Using Denavit-Hartenberg (DH) convention, we identify the axes of these three joints in the reference configuration:

$$\theta_1 \rightarrow z_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}^T$$
$$\theta_2 \rightarrow z_2 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$$
$$z_3 \rightarrow z_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$$

The forward kinematics (FK) from point P_O (origin) to P_E (end-effector) is found by multiplying the transforms given by the three joints. This effectively yields a mapping between the jointspace configuration to the end-effector position (operational space):

$$P_E = T_O^E P_O = T_{\theta 1} T_{\theta 2} T_{d3} = R_{z1}(\theta_1) R_{z2}(\theta_2) t_3(d_3)$$

Working out the trigonometric matrix calculations, we

find the full FK map as follows:

$$T_O^E = \begin{bmatrix} \frac{1}{2}\left(c_2(c_1+1)+s_2(c_1-1)\right) & -\frac{s_1}{\sqrt{2}} \\ \frac{s_1}{\sqrt{2}}(c_2+s_2) & c_1 \\ -\frac{1}{2}\left(c_2(c_1-1)+s_2(c_1+1)\right) & \frac{s_1}{\sqrt{2}} \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{2}\left(s_2(c_1+1)-c_2(c_1-1)\right) & \frac{1}{2}d_3\left(s_2(c_1+1)-c_2(c_1-1)\right) \\ -\frac{s_1}{\sqrt{2}}(c_2-s_2) & -\frac{s_1}{\sqrt{2}}d_3(c_2-s_2) \\ \frac{1}{2}\left(c_2(c_1+1)-s_2(c_1-1)\right) & \frac{1}{2}d_3\left(c_2(c_1+1)-s_2(c_1-1)\right) \\ 0 & 1 \end{bmatrix}$$

The inherent structure of the RRP manipulator only allows control of cartesian position in operational space (i.e. the orientation is dependent on the direction the boom points and is not independently controllable). Thus, we are particularly interested by the position of the end-effector given by the last column of the full FK transform in $SE(3)$:

$$\mathbf{p_E} = \begin{bmatrix} x_E \\ y_E \\ z_E \end{bmatrix} = \begin{bmatrix} \frac{1}{2}d_3\left(s_2(c_1+1)-c_2(c_1-1)\right) \\ -\frac{s_1}{\sqrt{2}}d_3(c_2-s_2) \\ \frac{1}{2}d_3\left(c_2(c_1+1)-s_2(c_1-1)\right) \end{bmatrix}$$

For the inverse kinematics mapping going from a cartesian position in operational space to joint coordinates, we find a trigonometric relation using spherical coordinate geometry as:

$$\mathbf{q} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} \arctan\left(\frac{-\sqrt{2}y}{-x+z}\right) \\ -\arccos\left(\frac{x+z}{\sqrt{2(x^2+y^2+z^2)}}\right)+\frac{\pi}{4} \\ \sqrt{x^2+y^2+z^2} \end{bmatrix}$$

Taking the partial derivatives of the end-effector position $\mathbf{p_E}$ with respect to joint coordinates allows us to find the linear velocity Jacobian:

$$J_v = \frac{\partial \mathbf{p}_E}{\partial \mathbf{q}} = \begin{bmatrix} \frac{\partial \mathbf{p}_E}{\partial \theta_1} & \frac{\partial \mathbf{p}_E}{\partial \theta_2} & \frac{\partial \mathbf{p}_E}{\partial d_3} \end{bmatrix}$$

$$J_v = \begin{bmatrix} -\frac{1}{2}d_3(s_2-c_2)s_1 & \frac{1}{2}d_3\left((c_2+s_2)c_1+c_2-s_2\right) \\ -\frac{c_1}{\sqrt{2}}d_3(c_2-s_2) & \frac{s_1}{\sqrt{2}}d_3(c_2+s_2) \\ -\frac{1}{2}d_3(c_2-s_2)s_1 & \frac{1}{2}d_3\left(-(c_2+s_2)c_1+c_2-s_2\right) \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{2}\left(s_2(c_1+1)-c_2(c_1-1)\right) \\ -\frac{s_1}{\sqrt{2}}(c_2-s_2) \\ \frac{1}{2}\left(c_2(c_1+1)-s_2(c_1-1)\right) \end{bmatrix}$$

Additionally, the angular velocity Jacobian J_ω can be found as the direction of the two revolute joint axes, given as:

$$J_\omega = \begin{bmatrix} \mathbf{z_1} & \mathbf{z_2} & 0 \end{bmatrix}$$

$$J_\omega = \begin{bmatrix} \frac{1}{2}(1-c_1) & s_2\frac{1}{2}(1+c_1)+c_2\frac{1}{2}(1-c_1) & 0 \\ -\frac{s_1}{\sqrt{2}} & s_2\frac{s_1}{\sqrt{2}}-c_2\frac{s_1}{\sqrt{2}} & 0 \\ \frac{1}{2}(1+c_1) & s_2\frac{1}{2}(1-c_1)+c_2\frac{1}{2}(1+c_1) & 0 \end{bmatrix}$$

3.2 Control Dynamics Model

Define the state vector as the combination of both joint coordinates $\mathbf{q}(t)$ and their respective velocities $\dot{\mathbf{q}}(t)$:

$$\mathbf{q}(t) = \begin{bmatrix} \theta_1(t) & \theta_2(t) & d_3(t) \end{bmatrix}^T$$

$$\dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{\theta}_1(t) & \dot{\theta}_2(t) & \dot{d}_3(t) \end{bmatrix}^T$$

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix}$$

Consider the control input as the three joint torques for the RRP manipulator:

$$\mathbf{u}(t) = \begin{bmatrix} \tau_1 & \tau_2 & f_3 \end{bmatrix}$$

In general manipulator theory, the second-order joint dynamics can be described as:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + G(\mathbf{q}) = \mathbf{u}$$

In our application, since we are concerned with safe trajectories involving relatively low velocities/accelerations, we neglect the Coriolis term $C(\mathbf{q}, \dot{\mathbf{q}}) \approx 0$ to simplify the dynamics formulation. Written in state-space representation, we have:

$$M(\mathbf{q})\ddot{\mathbf{q}} + G(\mathbf{q}) = \mathbf{u}$$

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \dot{\mathbf{q}} \\ M(\mathbf{q})^{-1}\left(\mathbf{u} - G(\mathbf{q})\right) \end{bmatrix}$$

The mass matrix itself is derived from the basic Jacobian as follows:

$$M(\mathbf{q}) = m_{boom}J_v^T J_v + J_\omega^T I_{boom} J_\omega$$

Note that the mass and inertia tensor of the boom effectively increase as the boom is extended further (dependent on d_3). Leveraging the linear density of the boom $\lambda_b = 0.12$ kg/cm and the constant mass of the end-effector $m_{ee} = 0.1042$ kg, we find the inertial terms of the boom as follows:

$$m_{boom}(d_3) = m_{ee} + \lambda_b d_3$$

$$I_{boom}(\mathbf{q}) = \left(\frac{1}{3}\lambda_b d_3 + m_{ee}\right)\begin{bmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{bmatrix}$$

The gravity vector is subsequently computed from the gravitational potential energy:

$$\begin{aligned} V(\mathbf{q}) &= \left(\frac{1}{2}\lambda_b d_3 + m_e e\right) g z_E \\ &= \left(\frac{1}{2}\lambda_b d_3 + m_e e\right) g \left(\frac{1}{2}d_3\left(c_2(c_1+1)-s_2(c_1-1)\right)\right) \end{aligned}$$

$$\begin{aligned} G(\mathbf{q}) &= -\nabla_{\mathbf{q}} V(\mathbf{q}) = -\begin{bmatrix} \frac{\partial V}{\partial \theta_1} \\ \frac{\partial V}{\partial \theta_2} \\ \frac{\partial V}{\partial d_3} \end{bmatrix} \\ &= -\frac{1}{2}\left(\frac{1}{2}\lambda_b d_3 + m_{ee}\right) g \begin{bmatrix} d_3\left(-c_2 s_1 + s_2 s_1\right) \\ d_3\left(-s_2(c1+1) - c_2(c_1-1)\right) \\ \left(c_2(c_1+1) - s_2(c_1-1)\right) \end{bmatrix} \end{aligned}$$

We thus have a complete model of the boom manipulator's dynamics $\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u})$. We will proceed with an affine approximation (linearization) of this highly non-linear dynamics model with our subsequent sequential convex programming (SCP) approach.

4 Optimal Trajectory Generation

4.1 Overview

We begin by considering generation of an optimal open-loop trajectory between initial and goal end-effector positions in operational space. It is easier to define these states initially in Cartesian coordinates $\mathbf{p_E} = \begin{bmatrix} x_E & y_E & z_E \end{bmatrix}^T$ then use the inverse kinematics transform to map to jointspace coordinates $\mathbf{q} = \begin{bmatrix} \theta_1 & \theta_2 & d_3 \end{bmatrix}^T$. We outline appropriate formulations for the cost and constraints in the following sections to maximize performance and safety. Then, a sequential convex programming (SCP) implementation is used to iteratively solve the nonlinear optimization problem for trajectory generation.

4.2 Cost Formulation

To keep our problem simple, we consider a quadratic cost in both state and control effort. For control effort, we hypothesize that minimizing shoulder torques (torque at joints θ_1, θ_2) should take priority over minimizing boom force. In practice, this is mechanically realistic since it is harder to design super strong shoulder joints to support the loads on the cantilevered boom compared to generating a larger linear force with a boom deployer. We reflect this in the quadratic term:

$$\sum_{k=0}^{N-1} u_k^T R u_k$$

$$= \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.001 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

For the states, we wish to encourage the boom to move towards the goal state. Here, positional accuracy of the revolute shoulder joints is of greater significance than the accuracy of the prismatic boom since the orientation of our "reaching" direction is critical during manipulation tasks (i.e. reaching towards a rock). Thus, we formulate the following quadratic state cost:

$$\sum_{k=0}^{N-1} \left(x_k - x_{\text{goal}}\right)^\top Q \left(x_k - x_{\text{goal}}\right)$$

$$Q = \begin{bmatrix} 0.01 & & & & & \\ & 0.01 & & & & \\ & & 0.001 & & & \\ & & & 0.001 & & \\ & & & & 0.001 & \\ & & & & & 0.0001 \end{bmatrix}$$

Through preliminary trial and error, we observe that having Q too large promotes overly aggressive initial maneuvers toward the goal position (at the expense of higher control efforts). Thus, we make Q an order of magnitude smaller than R to prioritize minimization of control efforts.

Lastly, we consider a terminal state cost term to replace the use of an explicit constraint $x_N = x_{goal}$ on the terminal state.

$$\left(x_N - x_{\text{goal}}\right)^\top P \left(x_N - x_{\text{goal}}\right)$$

This strategy is useful in improving stability during each iteration of SCP (where the nominal trajectory may not necessarily reach the goal state initially). Thus, our overall cost function for minimization is defined as:

$$\begin{aligned} J(\mathbf{x}, \mathbf{u}) = &\left(x_N - x_{\text{goal}}\right)^\top P \left(x_N - x_{\text{goal}}\right) \\ &+ \sum_{k=0}^{N-1} \left(x_k - x_{\text{goal}}\right)^\top Q \left(x_k - x_{\text{goal}}\right) \\ &+ \sum_{k=0}^{N-1} u_k^T R u_k \end{aligned}$$

4.3 Constraint Formulation

Now we consider the necessary constraints to formulate a trajectory generation problem. Intuitively, we can define a state constraint at the initial starting position with zero velocity:

$$x_0 = x_{init} = \begin{bmatrix} \theta_{1,0} \\ \theta_{2,0} \\ d_{3,0} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

There is also evidently a state constraint from the discrete-time dynamics. In the prior section, we outlined the derivation of the complete nonlinear manipulator dynamics

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \dot{\mathbf{q}} \\ M(\mathbf{q})^{-1} \left(\mathbf{u} - G(\mathbf{q})\right) \end{bmatrix}$$

We then use a Runge-Kutta integration method to discretize these continuous dynamics.

$$x_{k+1} = f_d(x_k, u_k)$$

Then, we convert to a convex approximation by taking the affine linearization of the discrete dynamics around the nominal trajectory:

$$\begin{aligned} f(x_k, u_k) &\approx A_k s_k + B_k u_k + c_k \\ A_k &= \frac{\partial f_d}{\partial x}\left(x^{(i)}, u^{(i)}\right) \\ B_k &= \frac{\partial f_d}{\partial u}\left(x^{(i)}, u^{(i)}\right) \\ c_k &= f_d\left(x^{(i)}, u^{(i)}\right) - A_k x^{(i)} - B_k u^{(i)} \end{aligned}$$

This yields the following equality constraint capturing the linearized state dynamics:

$$x_{k+1} = A_k x_k + B_k u_k + c_k$$

Next, we consider the constraints arising from actuator limits. The current ReachBot hardware prototype uses Dynamixel XL430-W250-T motors, with a max stall torque of 1.5 N-m. Since the prismatic boom joint has a different max force, we consider the joints separately in formulating the constraints:

$$u_1,\ u_2 \leq \tau_{max}$$

$$f_3 \leq f_{max}$$

Finally, to ensure the validity of our linear approximation of the dynamics, we impose a trust region constraint to ensure we remain near the nominal trajectory used for linearization. This is defined as an inequality constraint for a box around the nominal trajectory:

$$\left\| x_k - x_k^{(i)} \right\|_\infty \leq 1, \quad \forall k \in \{0, 1, \ldots, N\}$$
$$\left\| u_k - u_k^{(i)} \right\|_\infty \leq 1, \quad \forall k \in \{0, 1, \ldots, N-1\}$$

4.4 Problem Statement

To summarize, the trajectory generation optimization using SCP with a convex approximation of dynamics is given as:

$$\begin{aligned} &\text{minimize} J(\mathbf{x}, \mathbf{u}) \\ &\text{subject to } x_0 = x_{init} \\ &\qquad x_{k+1} = Ax_k + Bu_k + c_k \\ &\qquad u_1, u_2 \leq u_{max} \\ &\qquad f_3 \leq f_{max} \\ &\qquad \left\| x_k - x_k^{(i)} \right\|_\infty \leq 1 \\ &\qquad \left\| u_k - u_k^{(i)} \right\|_\infty \leq 1 \end{aligned}$$

We discuss the performance of our optimization strategy with respect to tuning different parameters (such as the cost function and actuator limits) and provide a comparison with a classical PD control scheme in the following sections.

5 Trajectory Generation Experiments

We consider a Python implementation of the SCP optimization problem outlined in the prior section. As a note, we perform a minor alteration of our dynamics model due to several limiting factors in practice. (1) The actual hardware system with Dynamixel motors does not allow for direct torque control, so we work with velocity control instead. (2) The complex nature of our full manipulator dynamics (with several large matrices) led to several convergence issues in CVXPY (this may be addressed with future tweaking of solver parameters in the future). To compensate for these practical limitations, we change the control input to acceleration-based control in lieu of torque-based control.

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix} = \begin{bmatrix} \ddot{\theta}_1(t) \\ \ddot{\theta}_2(t) \\ \ddot{d}_3(t) \end{bmatrix}^T$$

The dynamics update subsequently becomes simplified as:

$$\ddot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} x_4(t) \\ x_5(t) \\ x_6(t) \\ u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix}$$

This simplified model is highly accurate in conditions where gravity is less significant (such as microgravity environments in space), and where the inertial (mass) effects of the boom do not need to be fully captured with the full dynamics model. Given additional time and computational resources, generalizing to the more complex manipulator dynamics is straightforward and should yield similar (although not identical) results.

5.1 Initial Trajectory Results

We perform SCP with the previously discussed trajectory generation problem. The values of $P,\ Q,\ R$ are consistent with the aforementioned matrices, while the dynamics utilize our simplified acceleration-based control scheme as an initial baseline. The trust region constraint and actuator limits are also maintained as previously discussed. The initial trajectory we consider is a "reaching" motion with initial and goal end-effector positions in Cartesian space as:

$$p_{init} = \begin{bmatrix} 0 & 0 & 0.05 \end{bmatrix}^T \text{ m}$$
$$p_{goal} = \begin{bmatrix} 0.5 & 0.5 & 1.5 \end{bmatrix}^T \text{ m}$$

Running on a 2022 M2 Macbook Air, we find SCP convergence within 10 iterations for all test cases, with runtimes of around 7-10 seconds. The fast convergence is most likely a result of the simplified dynamics we utilize in this initial approach.

We are interested in studying the effect of the actuator limit (how much maximal force/acceleration we allow the joints to experience) on the particular trajectories generated. Thus, a parametric study is performed on different levels of actuator limit (strict, moderate, lenient). Interestingly, there is a tradeoff between boom extension (d_3 effort) and gimbal rotation ($\theta_1,\ \theta_2$ effort) as we tweak the actuator limits.

To quantify the significance of an optimal control approach for our application, we also present a comparison with a simple PD controller for position control. We observe large unbounded control efforts with the PD controller, potentially resulting in dangerous and unfeasible trajectories when applied to the physical system.

5.2 Actuator Limit Effects

From a mechanical perspective, limits on the revolute shoulder joints are more critical for safety/stability. Thus, we consider a constant actuation limit on the prismatic joint fixed to be:

$$u_3 \leq 0.5 \text{ m/s}^2$$

Then, we consider three different levels of shoulder control effort limit, defined as strict, moderate, and lenient:

$$\text{[Strict:]}\ u_1,\ u_2 \leq 0.025 \text{ rad/s}^2$$
$$\text{[Moderate:]}\ u_1,\ u_2 \leq 0.075 \text{ rad/s}^2$$
$$\text{[Lenient:]}\ u_1,\ u_2 \leq 0.25 \text{ rad/s}^2$$

The resulting 3D trajectories from our SCP optimization between the specified initial/goal positions are shown in Fig. 3.

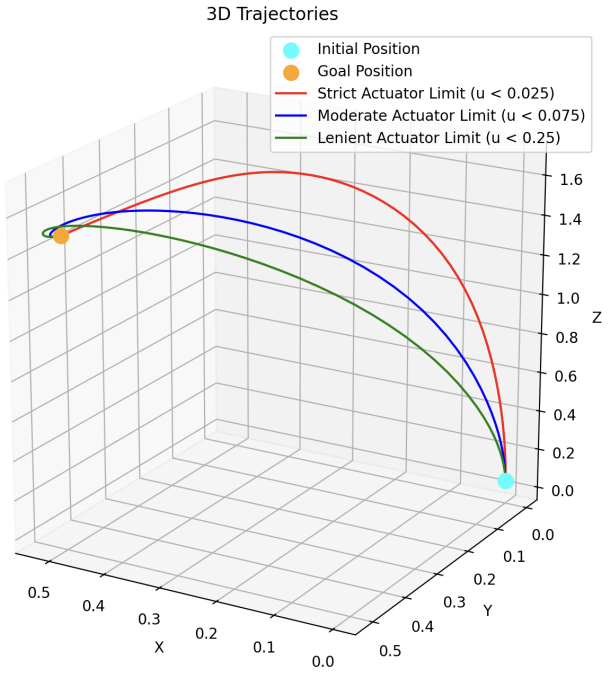

Fig. 3: 3D end-effector trajectories from SCP optimization. Three different levels of actuator limit on the revolute joints ($\theta_1,\ \theta_2$) are considered.

We observe a distinct tradeoff in approach between the three trajectories based on how strict of an actuator limitation we impose. In the case of a lenient limit ($u_1,\ u_2 \leq 0.25$) we observe an immediate and rapid reconfiguration of the shoulder joint to point toward the goal position as the boom extends. However, with a strict actuator limit ($u_1,\ u_2 \leq 0.025$) we see more modest and small shoulder movements initially (as the boom begins to extend), then a more gradual reorientation towards the goal. While reorientation of the boom when it is more extended (longer) may be unfavorable due to the larger manipulator inertias involved, this heuristic may not be fully captured in our current approach due to either the simplified dynamics or simple quadratic cost formulation.

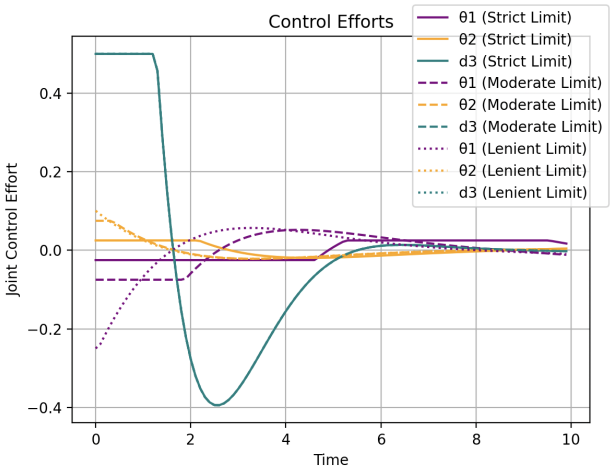

Fig. 4: Control effort trajectories from SCP optimization. Three different levels of actuator limit on the revolute joints ($\theta_1,\ \theta_2$) are considered.

We examine the actual control inputs over time in Fig. 4 to ensure our actuation constraints are successful. Since the same actuator limit is used in all trials for the prismatic joint (d_3), we see identical behavior in the u_3 control effort. For the lenient limit ($u_1,\ u_2 \leq 0.25$), we see that the revolute shoulder joints do not saturate at the limit, and are thus purely optimized with respect to the quadratic cost term. The moderate and strict limits ($u_1,\ u_2 \leq 0.075,\ 0.025$) exhibit control saturation at the actuator limits during the first 2 seconds of the trajectory, but are able to stabilize regardless afterward.

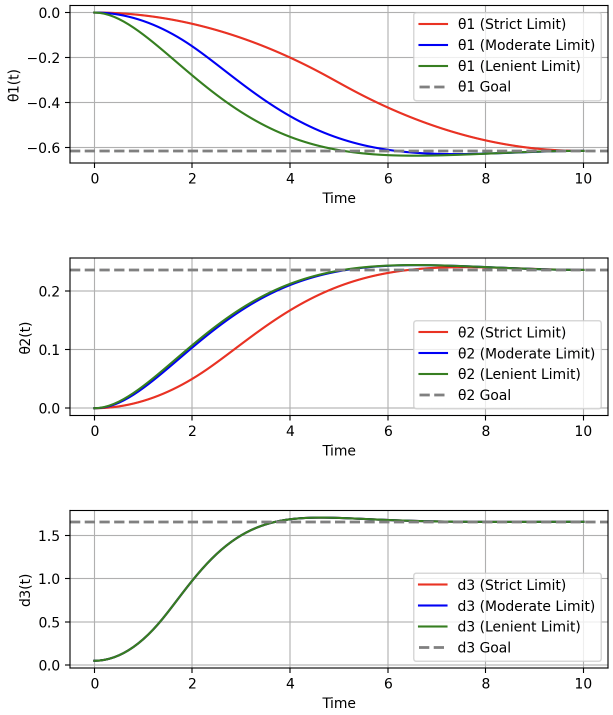

Fig. 5: Joint trajectories ($\theta_1,\ \theta_2,\ d_3$) from SCP optimization. Three different levels of actuator limit on the revolute joints ($\theta_1,\ \theta_2$) are considered.

Finally, we also consider the actual trajectories of the three joints in Fig. 5 to access stability. Reasonably, we observe that having more lenient actuator limits allows for more aggressive initial trajectories that converge faster toward the goal. Interestingly, all three joints exhibit a minor amount of overshoot past the goal value midway through the trajectory, before compensating and returning to the goal. This motivates the potential use of a different controller (such as PID) to stabilize the system around the goal position once the optimal trajectory brings it close enough.

5.3 Comparison with PD Control

To further motivate the significance of an optimal control framework for trajectory generation, we compare our results to the performance of a classical PD (proportional-derivative) control scheme. Using a PD controller, the control effort at each joint is simply proportional to the feedback of an error term and its derivative:

$$u_1 = K_p(\theta_{1,goal} - \theta_1) - K_v\dot{\theta}_1$$
$$u_2 = K_p(\theta_{2,goal} - \theta_2) - K_v\dot{\theta}_2$$
$$u_3 = K_p(d_{3,goal} - d_3) - K_v\dot{d}_3$$

Through some trial and error, we find that tuning the gains to around $K_p = K_v = 5$ yielded stable performance with comparable settling time to the 10 second window of our SCP framework. We subsequently proceed with comparing the performance of the PD control scheme to the SCP optimization approach with moderate actuator limit $(u_1,\ u_2 \leq 0.075)$. The resulting end-effector trajectories are plotted in Fig. 6.

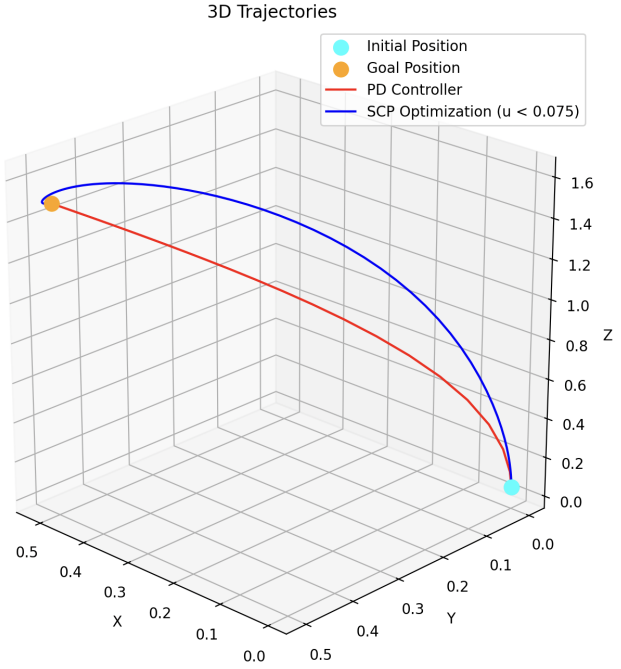

Fig. 6: 3D end-effector trajectories from (1) PD control (2) SCP optimization

Since there are no actuator limits on PD control, we see the trajectory almost immediately drive toward the goal point with strong initial revolute joint efforts (compared to the optimal framework with actuator limits). This aggressive maneuver could be lessened by tuning separate gains for each joint of our manipulator (or perhaps even using a controller in operational space), but the lack of explicit constraints on control effort remains a limitation.

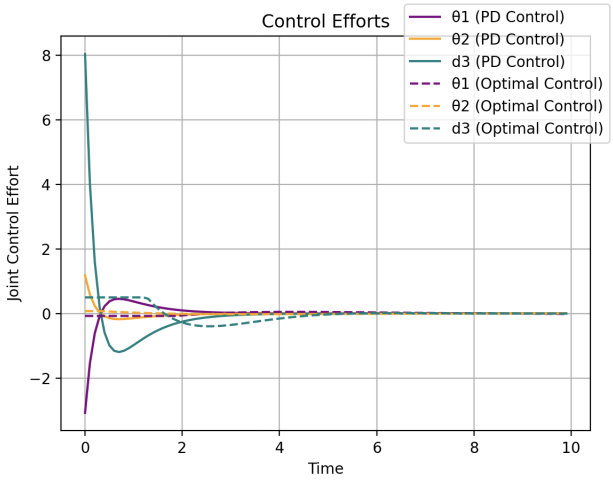

Fig. 7: Control effort trajectories from (1) PD control (2) SCP optimization

We subsequently compare the actual control efforts experienced between PD and our optimal control methods in Fig. 8. Unsurprisingly, we see enormous initial control efforts with the PD controller (when error is large), indicated by the solid lines on the plot. This behavior would be highly unsafe in implementation since it may result in excessively large forces/accelerations resulting in system failure. On the other hand, the control efforts from our SCP approach remain bounded by constraints, and remain far smaller than PD control throughout the 10 second horizon.

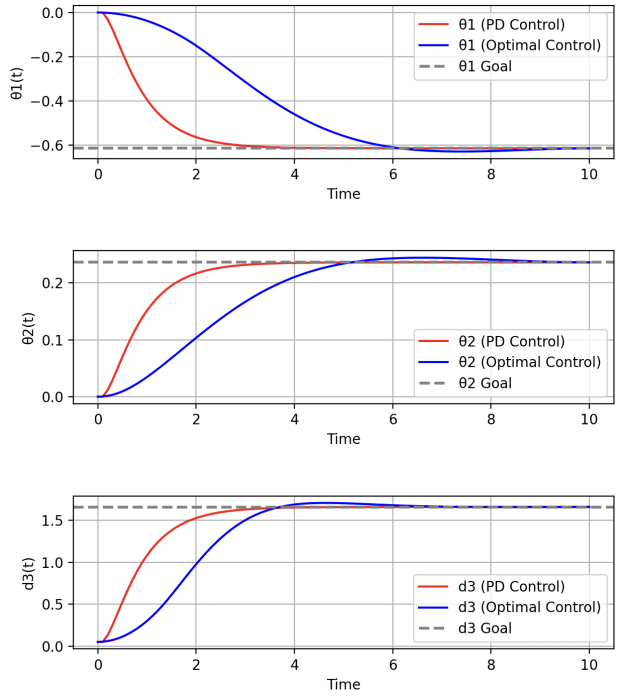

Fig. 8: Joint trajectories $(\theta_1,\ \theta_2,\ d_3)$ from (1) PD control (2) SCP optimization

Finally, we consider the joint trajectories observed between PD control and optimal control. The settling time of our PD system is slightly faster, leading to all joints converging to the goal in less time. Furthermore, PD does not exhibit overshoot around the goal, so it may be desirable for stabilizing when near the goal pose. For future exploration, it would be interesting to adjust the time horizon of our optimization problem (to less than 10 seconds) to encourage faster settling on par with PD control–in which case the control efforts become more directly comparable.

6 Hardware Experiments

To validate the feasibility of our trajectories, we perform initial tests in applying pre-generated optimal trajectories to our hardware system. As a simple demonstration task, we concern ourselves with planning a trajectory steering the boom from a retracted initial state towards successfully grasping a rock. Taking some measurements in the lab, we define the requisite initial and goal states for this problem as:

$$p_{init} = \begin{bmatrix} 0 & 0 & 0.15 \end{bmatrix}^T \text{ m}$$

$$p_{goal} = \begin{bmatrix} -0.01 & -0.07 & 0.95 \end{bmatrix}^T \text{ m}$$

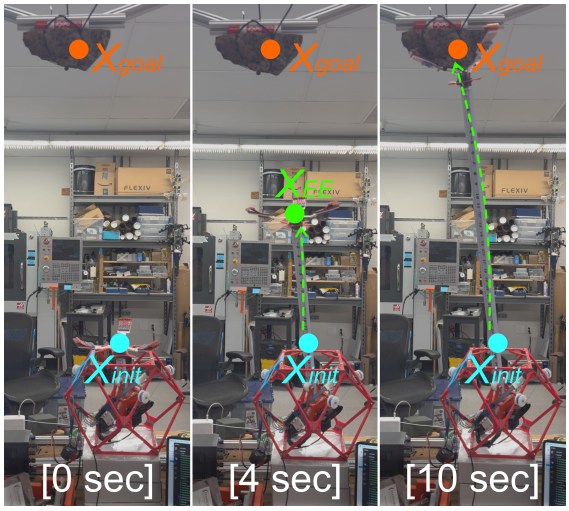

Fig. 9: Snapshots of real end-effector position for a rock grasping task following an optimal trajectory generated using our SCP framework

Fig. 9 shows the end-effector trajectory of our hardware system following a pre-generated trajectory from our SCP planner. Over 10 trials, we observe a high degree of repeatability for the hardware system to follow the SCP trajectory, with the end-effector contacting (but not necessarily grasping) the rock in all trials. The success rate of the rock grasping task is around 50%, primarily limited by the performance of the hardware system (inability of gripper to activate and grasp successfully). While a real-time closed-loop policy (such as an MPC planner) could potentially achieve better results, we nevertheless see an acceptable level of performance in simpler tasks with the robot's ability to follow optimal open-loop trajectories.

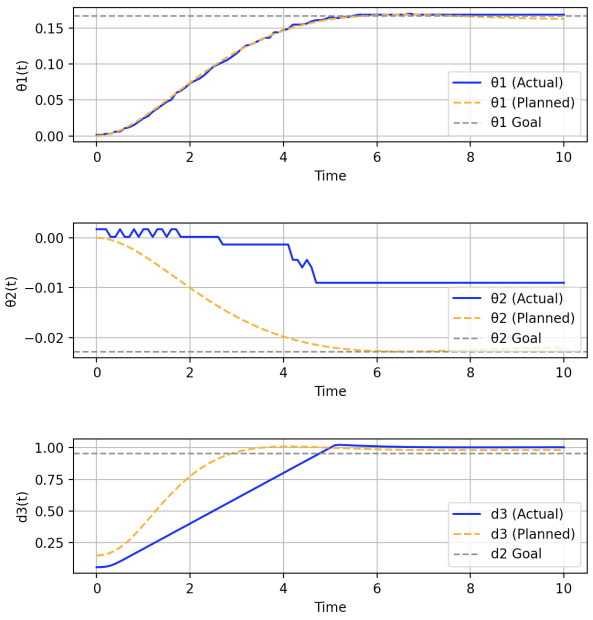

Fig. 10: Joint trajectories $(\theta_1,\ \theta_2,\ d_3)$ measured from actual hardware compared to nominal trajectories generated from the open-loop SCP planner

Fig. 10 provides a detailed comparison of actual joint trajectories realized on hardware and the nominal joint trajectories from the SCP planner. For joint θ_1, we see an overall displacement of around 0.15 radians throughout the trajectory, with excellent correspondence between the physical hardware and nominal trajectory. Joint θ_2 experiences a much smaller displacement of around 0.02 radians (almost negligible movement), and thus the discrepancy between hardware and nominal is not significant. Finally, the prismatic joint d_3 appears to experience velocity saturation on the actual hardware (hitting max velocity of the boom motor), leading to an inability to keep up with the nominal trajectory. A velocity limit on the boom deployment rate may need to be considered in future work to yield more feasible trajectories in this regard.

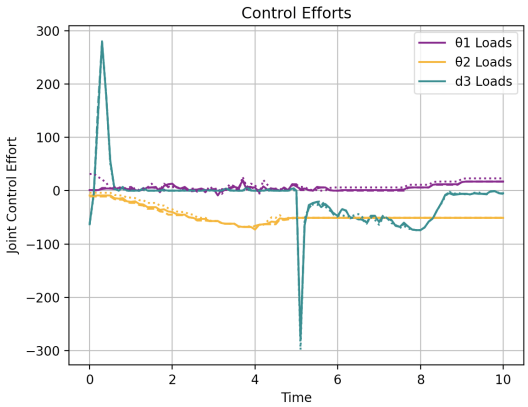

Fig. 11: Loads experiences at each joint motor in hardware when following planned optimal trajectories

Fig. 11 examines the physical loads experienced by our joint motors when following the prescribed optimal trajectories. Interestingly, we see sharp spikes in the boom actuator on startup and when contacting the rock, which are inherent to the physical nature of our application. Additionally, there is a load buildup on the second revolute joint θ_2, most likely due to the torque induced by the extending boom's increasing gravitational load.

7 Conclusions and Future Work

We present a novel application of optimal control for trajectory planning of an extendable boom manipulator for ReachBot. Using Sequential Convex Programming (SCP), we generate optimal end-effector trajectories, considering state constraints, actuator limits, and a trust region. Our findings show that actuator limits significantly influence trajectories, and the optimization-based approach outperforms PD control. Finally, we validate our planned trajectories on physical hardware with a rock-grasping task.

Several immediate extensions exist to increase the applicability of this work to our robot. While we present a complete formulation of the RRP manipulator dynamics for our boom, more work remains in properly integrating these complex dynamics in a linearized/discrete manner for optimization. Furthermore, more studies bridging the sim2real gap–studying proper shaping of cost and constraints to yield high performance on hardware–can also be considered. Finally, implementation of a real-time closed-loop policy (most likely MPC) could further enhance the robustness of our system in a wider range of tasks and conditions.

References

[1] Robert H. Jr. Cannon and Eric Schmitz. Initial experiments on the end-point control of a flexible one-link robot. *Journal of Spacecraft and Rockets*, 21(2):167–176, 1984.

[2] Tony G Chen, Stephanie Newdick, Julia Di, Carlo Bosio, Nitin Ongole, Mathieu Lapôtre, Marco Pavone, and Mark R Cutkosky. Locomotion as manipulation with reachbot. *Science Robotics*, 9(89):eadi9762, 2024.

[3] ZhongYi Chu and YiAn Lei. Design theory and dynamic analysis of a deployable boom. *Mechanism and Machine Theory*, 71:126–141, 2014.

[4] Matthew T. Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 14(2):210–222, 1984.

[5] Daniel Morton, Mark Cutkosky, and Marco Pavone. Task-driven manipulation with reconfigurable parallel robots. *arXiv preprint arXiv:2403.10768*, 2024.

[6] KA Seffen, S Pellegrino, and GT Parks. Deployment of a panel by tape-spring hinges. In *IUTAM-IASS Symposium on Deployable Structures: Theory and Applications: Proceedings of the IUTAM Symposium held in Cambridge, UK, 6–9 September 1998*, pages 355–364. Springer, 2000.

[7] Rex J. Theodore and Ashitava Ghosal. Modeling of flexible-link manipulators with prismatic joints. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 27(2):296–303, 1997.

———-