Hybrid Trajectory Optimization for Monkey Bar Robot Using DIRCOL

Judson Kyle[1], Hao Luo[1], David Ologan[1]

***Abstract*—In this paper we show a trajectory planning technique that mimics a monkey bar robot swinging from bar to bar. Using a hybrid system direct collocation (DIRCOL) trajectory optimization, we successfully demonstrate the robot swinging up from a dead hang to catch the first bar and swing to the subsequent bars. This DIRCOL technique was tested on various mass distributions in the robot as well as different bar separation distances to understand the behavior with varying parameters. In addition, we show the importance of a free time setup on the cost function in producing consistent feasible trajectories using this DIRCOL technique.**

Code for this project can be found here: https://github.com/ologandavid/MonkeyBarBot

***Index Terms*—Trajectory Optimization, Brachiation, Two-Bar Linkage, DIRCOL**

I. Introduction

In recent years, developing versatile locomotion systems has attracted significant interest among robotics researchers. Motivated by the desire to have robots operate in increasingly complex and unpredictable environments, the need for robots to react quickly and plan effective trajectories in a variety of conditions has become ever more important. Many researchers have attempted to solve this issue by mimicking the motion of living animals. Brachiation robots, inspired by the movement of primates, present an alternate form of locomotion that enables robots to traverse across large gaps, move through trees, or climb vertical surfaces. Using a combination of swinging and grasping, brachiation robots can access areas where legged, wheeled or articulated platforms might fail.

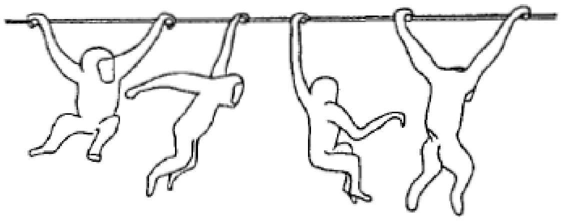

Fig. 1. Continuous Contact Brachiation of a Gibbon

With potential applications ranging from search and rescue, surveillance and reconnaissance, and the exploration of hazardous environments, a reliable trajectory planning algorithm capable of handling intermittent contacts is necessary. However, controlling the motion of brachiation robots remains a challenging problem given the complex nonlinear behavior of their kinematics as well as the particular handing of its hybrid modes.

To truly model the dynamics of a swinging primate, a 4-link robot with 6 actuators would be needed to accurately describe the wrist, elbow and shoulder joints. [1] Given the complexity of the system, we ignore the wrist joints and assume ”sticky” inelastic contacts with the bars. Additionally we simplify our dynamics by ignoring the elbow joints, considering only the motion of the shoulders. We find that these dynamics mirror those of a hybrid Acrobot. As such we propose a simplified dynamics model of a two link brachiation robot, and an associated direct collocation (DIRCOL) based approach for its trajectory planning. We verify its performance in simulation using MeshCat and Julia libraries, on a variety of bar orientations.

This paper aims to present the following.

1) Review the current state of the art in brachiation robotics
2) Present a simplified dynamics model of a two link brachiation robot
3) Introduce a DIRCOL based approach for trajectory optimization
4) Analyze the impact of various hyperparameters on the solved trajectory

The remainder of this work is organized as follows. Sec. II summarizes the related works. Sec. III-A, B describes a derivation of the kinematics of a simple two link robot, while Sec III-C describes our DIRCOL based approach for trajectory optimization. Subsequently, simulated results and solved trajectories are presented in Sec. IV, and V.

II. Related Work

A. Other Brachiation Platforms

Prior work has demonstrated other approaches at building and simulating brachiating robots. One of the earliest renditions of these robots was in fact a two-link robot, with a singular control input between the links. Utilizing a feedback control method that planned for continuous regular contact, the robot could traverse a course of evenly spaced bars. [2] Other brachiator robots like the JPL Brachiation Bot built on these advances, incorporating two additional links capable of traversing interspersed ledges. It implemented a four-phase hybrid movement, including a release, body reversal, swing up, and grasping phase. [3] Limitations in hardware due to structure of the grippers and issues with handling contact have also led researchers to pursue simulation based approaches. [4] [5] In response, other researchers have modeled the dynamics of swinging robots by treating contacts with the bar as distinct events, using reset maps to correct the dynamics after grabbing. [6]

B. Acrobot

After simplifying our dynamics, to those of a two link robot, we find that the dynamics during each swing phase resemble those of a double pendulum. These dynamics, are similar to those of a gymnast, with the top link pinned at bar, and a freely rotating second joint between the links. A more detailed analysis of these dynamics can be found here. [7] [8] This simplification allows the robot to move its shoulder to generate momentum to swing, with the pinned joint mimicking the hand in contact with the bar, and the free joint mimicking the hanging end of the pendulum. [1]

C. Control with DIRCOL

Trajectory Optimization for brachiating robots through complicated terrain has been approached using a variety of processes. Most recently a variation of iLQR and a PID tracking controller to was used to find the optimal trajectory of each swing phase allowing it to traverse bars with a variation in spacing. The method was even used to inform the robots' design by comparing the cost among different body lengths. [9] More traditional robust control Lyapunov methods have also been applied, incorporating barrier functions to handle the constraints and dynamic uncertainties of the motion. [10] AcroMonk, a two link brachiator, showed success when utilizing TVLQR, PD control, and reinforcement learning based policies in its control. [11] More complex approaches like NMPC have also been implemented, in favor of its ability to directly incorporate the constraints directly into the optimization problem. [12] Direct Collocation methods, summarized here [13] hold unique promise in their ability to yield dynamically feasible trajectories, but suffer in their handling of contact constraints.

III. Methods

A. Dynamics

We model Monkey Bar robot as a two-link pendulum, where the body mass is considered negligible compared to mass and inertia of both links. The variables in the dynamics are defined as in Fig.III-A.

We model the dynamics as piece-wise continuous. In between the bars, only one arm holds on a bar and the dynamics as continuous. After reaching the next bar, a reset map is applied to transit into the next phase. The spacing between bars is assumed to be varying for generalized results but the height of each bar is assumed to be constant.

a) Continuous Dynamics:

We derived the dynamics of the continuous phase based on the assumption that one arm is always grabbing on the bar, so there's no need to set up constraints on the end of the arms. Starting from the kinetic energy and potential energy terms in Lagrange equation:

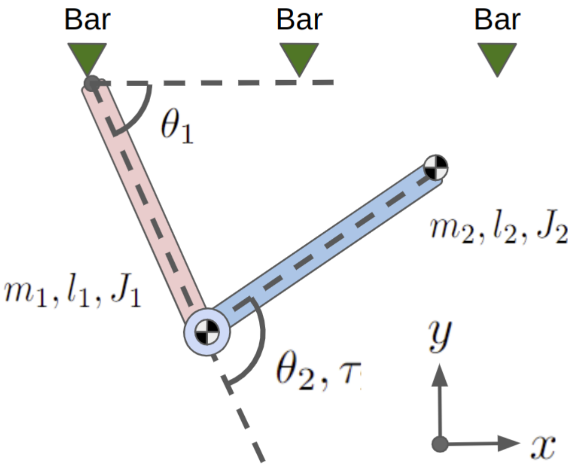

Fig. 2. Variable definition

$$\begin{aligned}
L =&T - V, \\
T =&\frac{1}{2}m_1(\dot{\theta}_1 l_1)^2+\frac{1}{2}m_2[(\dot{\theta}_1 l_1)^2+l_2(\dot{\theta}_1+\dot{\theta}_2)^2] \\
&+ m_2 l_1 l_2 cos(\theta_2)(\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_1^2) + \frac{1}{2}J_1\dot{\theta}_1^2 + \frac{1}{2}J_2\dot{\theta}_2^2 \\
V =&m_1 g l_1 sin(\theta_1) + m_2 g[l_1 sin(\theta_1) + l_2 sin(\theta_1 + \theta_2)]
\end{aligned}$$

The Lagrange equation gives:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{\theta}}) - \frac{\partial L}{\partial \theta} - \Upsilon = 0 \tag{1}$$

Each of the terms looks like:

$$\frac{\partial L}{\partial \theta} = \begin{bmatrix} (m_1+m_2)gl_2cos(\theta_1)+m_2gl_2cos(\theta_1+\theta_2) \\ m_2gl_2cos(\theta_1+\theta_2)-m_2l_1l_2sin(\theta_2)(\dot{\theta}_1\dot{\theta}_2+\dot{\theta}_1^2) \end{bmatrix} \tag{2}$$

$$\begin{aligned}
\frac{\partial L}{\partial \dot{\theta}_1} &= (m_1+m_2)l_1^2\dot{\theta}_1+m_2l_2^2(\dot{\theta}_1+\dot{\theta}_2) \\
&+ m_2l_1l_2cos(\theta_2)(2\dot{\theta}_1+\dot{\theta}_2) \\
\frac{\partial L}{\partial \dot{\theta}_2} &= m_2l_2^2(\dot{\theta}_1+\dot{\theta}_2)+m_2l_1l_2cos(\theta_2)(\dot{\theta}_1)
\end{aligned}$$

$$\begin{aligned}
\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_1} =&(m_1+m_2)l_1^2\ddot{\theta}_1+m_2l_2^2(\ddot{\theta}_1+\ddot{\theta}_2) \\
&+m_2l_1l_2cos(\theta_2)(2\ddot{\theta}_1+\ddot{\theta}_2) + m_2l_1l_2sin(\theta_2)(\dot{\theta}_1\dot{\theta}_2+\dot{\theta}_2^2)
\end{aligned} \tag{3}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_2} =m_2l_2^2(\ddot{\theta}_1+\ddot{\theta}_2)+m_2l_1l_2cos(\theta_2)(\ddot{\theta}_1) +m_2l_1l_2sin(\theta_2)(\dot{\theta}_1\dot{\theta}_2) \tag{4}$$

Collecting all the terms from Eq.2-4, we could write the entire dynamics as:

$$0 = M \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + B + G - \Upsilon \tag{5}$$

$$\Rightarrow \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = M^{-1}(\Upsilon - B - G) \tag{6}$$

$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{bmatrix} \tag{7}$$

$$m_{11} = m_1 l_1^2 + m_2[l_1^2 + l_2^2 + 2l_1 l_2 cos(\theta_2)] + J_1 + J_2$$

$$m_{12} = m_2[l_2^2 + l_1 l_2 cos(\theta_2)] + J_2$$

$$m_{22} = m_2 l_2^2 + J_2$$

$$B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} -l_1 l_2 m_2 sin(\theta_2)(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2) \\ -l_1 l_2 m_2 sin(\theta_2)\dot{\theta}_1^2 \end{bmatrix} \tag{8}$$

$$G = \begin{bmatrix} (m_1 + m_2) g l_2 cos(\theta_1) + m_2 g l_2 cos(\theta_1 + \theta_2) \\ m_2 g l_2 cos(\theta_1 + \theta_2) \end{bmatrix} \tag{9}$$

$$\Upsilon = \begin{bmatrix} 0 \\ \tau \end{bmatrix} \tag{10}$$

b) Reset Map:

To complete the model of the system, a reset map is needed to transition from the final state of one phase to the initial state of the next phase. This is accomplished through the reset map shown below.

$$g(X) = \begin{bmatrix} -(\pi - X[1]) \\ -X[2] \\ 0 \\ 0 \end{bmatrix} \tag{11}$$

Where X is the final state of the previous phase.

This reset map allows for use of a single dynamics equation thus reducing the complexity of the problem. The relation between initial and final states of the current and previous phases respectively were derived using trigonometry.

B. Constraints

For feasible trajectories, three sets of constraints were necessary. The first of these constraints is the primal bounds on the state variables and control inputs. No constraints were necessary for the state variables. Constraints on the time step were necessary, however, to ensure that the final result was feasible. The upper and lower bounds on this time step were set to be half of the goal time step and 2 times the goal time step respectively.

Equality constraints were added for initial conditions, terminal conditions, and dynamic feasibility. Initial and final condition constraints were used to constrain the problem to start and end in specific configurations described above in Reset Map. can be found using trigonometry to calculate the exact angles needed for initial and final states depending on bar separation and link length.

Since this problem can be converted into many separate sup-problems that have simple pendulum dynamics, the only dynamics constraints needed are for the 2-link pendulum problem. The jump map is only necessary for calculating the initial conditions for the following sub=problem trajectory optimization once the previous sub-problem has solved.

Constraining the final velocities to be zero is also critical for making this problem dynamically feasible. Under this assumption, we can neglect reaction forces once contact is made with the next bar thus reducing the complexity of the jump maps from final state of one sub-problem to initial state of the next.

C. Trajectory Optimization

A hybrid system direct collocation (DIRCOL) approach was used for trajectory optimization. For the hybrid systems analysis, the problem was broken up into 2 sub-problems that could be solved independently. The problem was divided into a swing up phase, where the robot starts from a dead hang, and a bar to bar phase, where the robot starts hanging on both bars, swinging to the next bar. These two trajectories are pictured in Fig.3.

Two conditions are required for this problem setup to be valid: (1) the initial conditions of the bar to bar phase have to be the final state of the previous phase and (2) the final conditions for all phases have to have zero angular velocity.

Condition (1) can be satisfied through the reset map shown above. This reset map is only valid for the case where the bar height remains constant. Without keeping this height constant, inverse trigonometric functions are required to solve for the transition angles which can result in DIRCOL failing to find a feasible trajectory.

Condition (2) removes the need to calculate reaction forces at the bar during contact. This reduces the complexity of the dynamics functions and thus simplifies the problem as a whole.

For this DIRCOL implementation, a simple quadratic cost function was used shown in (8). A simple cost on the state was used to encourage smaller magnitude angles in the final solution. This was also utilized for the input torque as well. In order to allow the solver to fit the natural pendulum dynamics, the time step for each iteration was also included as an input to the system with a reference cost placed on it. This reference cost placed a penalty on how far away the solver strayed from some goal time step.

Shown below is the final equation used for this trajectory tracking approach:

$$\begin{aligned} \min_{x,u} \quad & \frac{1}{2} x_N^T Q_f x_N + \Bigg[\sum_{k=1}^{N-1} \frac{1}{2} x_k^T Q x_k \\ & + \frac{1}{2}\Bigg(u_k - \begin{bmatrix} 0 \\ dt_{goal} \end{bmatrix}\Bigg)^T R \Bigg(u_k - \begin{bmatrix} 0 \\ dt_{goal} \end{bmatrix}\Bigg)\Bigg] \end{aligned} \tag{12}$$

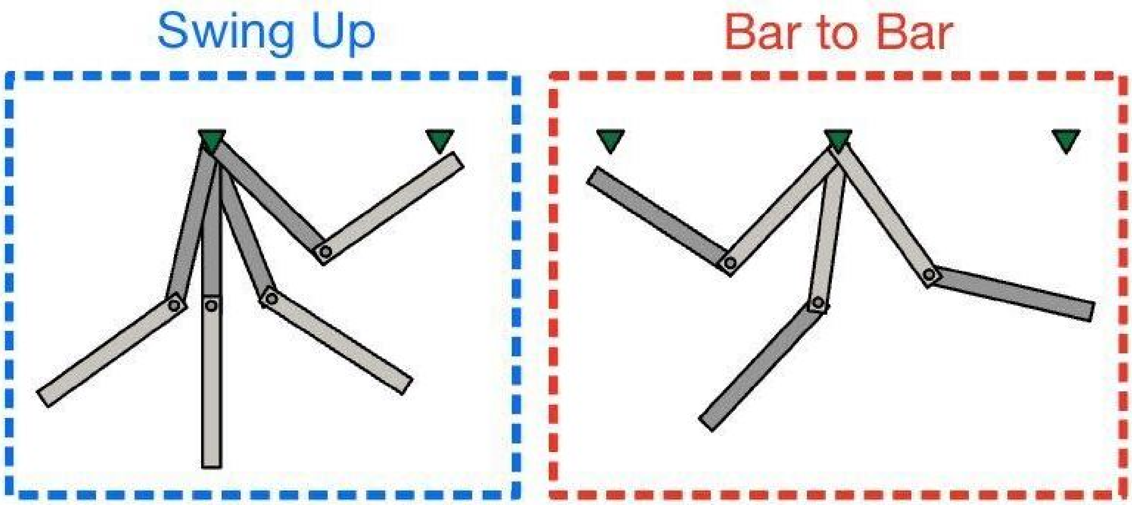

Fig. 3. Breakdown of hybrid trajectory

Subject To:

$$x_{k+1} = f(x_k, u_k) \quad \text{for } i = 1, 2, \ldots, N-1 \tag{13}$$

$$x_0 = x_{init} \quad \text{(Initial Condition)} \tag{14}$$

$$x_N = x_{goal} \quad \text{(Terminal Condition)} \tag{15}$$

$$u_{lb} \leq u_k \leq u_{ub} \quad \text{(Torque Limits)} \tag{16}$$

Where Q, Q_f, and R are the cost matrices discussed previously.

D. Implementation

For this implementation of DIRCOL, an IPOPT solver was used inside of the following pseudocode:

Algorithm 1 Monkey Bar Trajectory Optimization

```
Require: Robot dynamics $f$, control weight matrices $Q, R, Qf$, sub-problem parameters $N_{sub-problem}$, $Bar_{goal}$
Initialize dynamics, etc. parameters
for $i = 1 : N_{Bars}$ do
    if Starting configuration then
        $X_{ic} \leftarrow$ Dead Hang
    else
        $X_{ic} \leftarrow g(X_{prev}[end])$ {Reset Map}
    end if
    $X_g \leftarrow calculateGoalAngle(barSeparation)$
    $x_l, x_u \leftarrow$ primal bounds
    $z_0 \leftarrow$ Initial Guess Trajectory {zeros + noise}
    $Z_i \leftarrow$ Solve Dircol Trajectory Optimization
    $X_i, U_i, t_i \leftarrow Z_i$
    $X, U, t \leftarrow X_i, U_i, t_i$
end for=0
```

This algorithm was repeated for each different set of initial conditions tested.

An iPOPT solver was used for this DIRCOL implementation taking in the cost function, equality constraints, inequality constraints, and bounds on all variables.

IV. Results

We implemented the afore-mentioned optimization method in Julia and tested the generated trajectory in simulation. Fig.5 shows the full trajectory with bars evenly spaced away from each other at $1.5m$. Fig.6 shows the joint angles during this motion. Fig.7 shows the joint torque from actuator. In addition to the nominal trajectory, the solver was also used to test varying mass ratios and bar lengths across the horizon. Table shows the parameters for the various tests.

As shown in Fig.5, the robot successfully traverses the set of 5 bars starting from a dead hang. Also depicted in Fig.5 is the swing up behavior for this particular bar spacing and mass ratio. The optimal swing up trajectory for these conditions follows very closely the natural frequency of the fully extended double pendulum.

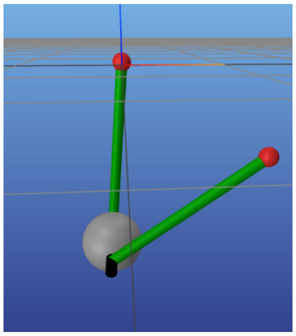

Fig. 4. Animation of monkeybot in motion in meshcat

Once this initial simulation was working, tests were performed with varying bar distance and mass ratios between feet and hip joints. Table I shows the parameters used for testing

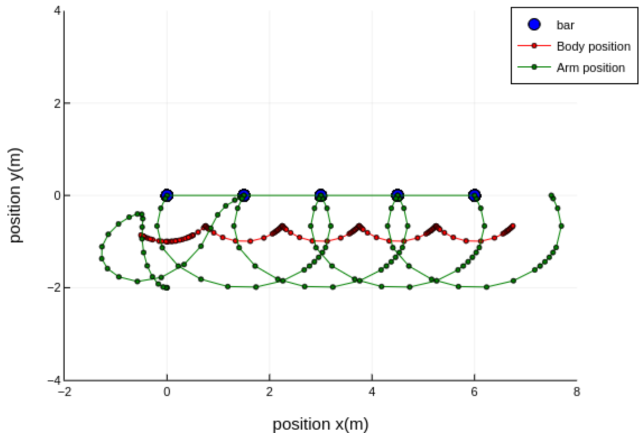

Fig. 5. Monkeybot full trajectory

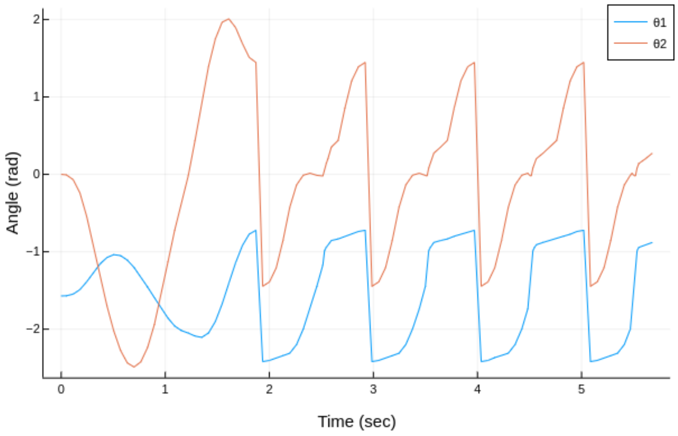

Fig. 6. Joint angles vs. Time

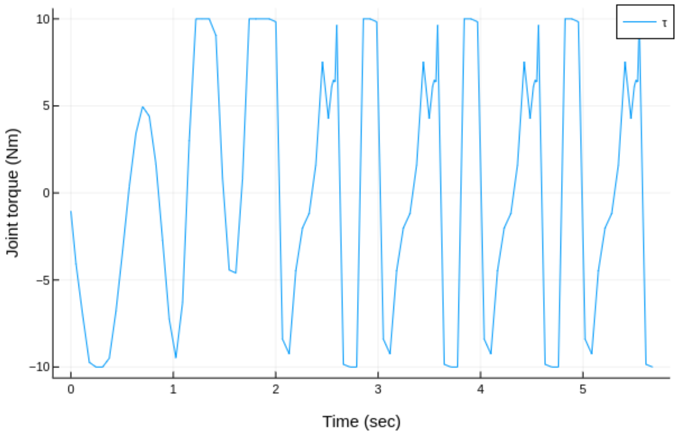

Fig. 7. Joint torque vs. Time

varying mass ratios. Fig.8 shows the results from this mass ration test in terms of joint angles over time. Table II shows the parameters used for varying bar separation distance. Fig.9 shows the results of this varying bar separation test in terms of joint angles over time.

Test No.	Mass 1	Mass 2
Test 1	1	1
Test 2	2	1
Test 3	5	1
Test 4	1	2

TABLE I
TESTING PARAMETERS FOR VARYING MASS RATIO TESTS

Bar Separation					
Test No.	Bar 1	Bar 2	Bar 3	Bar 4	Bar 5
Test 1	1.1	1.1	1.1	1.1	1.1
Test 2	1.5	1.5	1.5	1.5	1.5
Test 3	1.8	1.8	1.8	1.8	1.8

TABLE II
TESTING PARAMETERS FOR VARYING BAR SEPARATION

During testing, cost function was found to be critical to the solver converging to a feasible solution. Many variants of the cost function described above were tried including minimum time variants that attempted to minimize the time taken for this swing action to occur. While some feasible trajectories were produced, it was not consistent with the solutions and seemed to be very volatile to initial conditions. The free time implementation used for the final round of testing resulted in the best convergence rate and the most reasonable results.

In addition to the cost function itself, the weighting matrices also had a large effect on the feasibility and sensibility of the DIRCOL results. These weighting matrices were delicately tuned until reliable convergence was reached. The final weights that produced consistent feasible results placed higher weight on time step drift and state value rather than input torque.

V. DISCUSSION

Examining the results of the weighting matrices, having a higher cost on the time step drift allowed the solver to choose a time step that fit the natural dynamics of the double pendulum without going too far away from the goal time step. This resulted in smoother trajectories. Critical to this convergence was also the constraints on the time step itself. These constraints prevented the solver from ”cheating” physics and choosing time steps that were physically impossible but numerically possible.

In addition to cost function, the mass ratio between the hip (mass 1) and the feet (mass 2) also had a large effect on the resulting solution. When increasing the mass at the hip joint relative to the feet, the resulting trajectory reflected more closely that of the fully extended double pendulum. As shown in Fig.8, the mass ratio of $5:1$ had much smaller theta 1 peaks than each of the other tests increasing to a maximum when the mass ratio was $1:2$. In addition, Fig.8 demonstrates the same phenomenon where θ_2 is used more to move the system when the mass at the hip is larger. This fits with the way the weighting of the states was set up to penalize input torque.

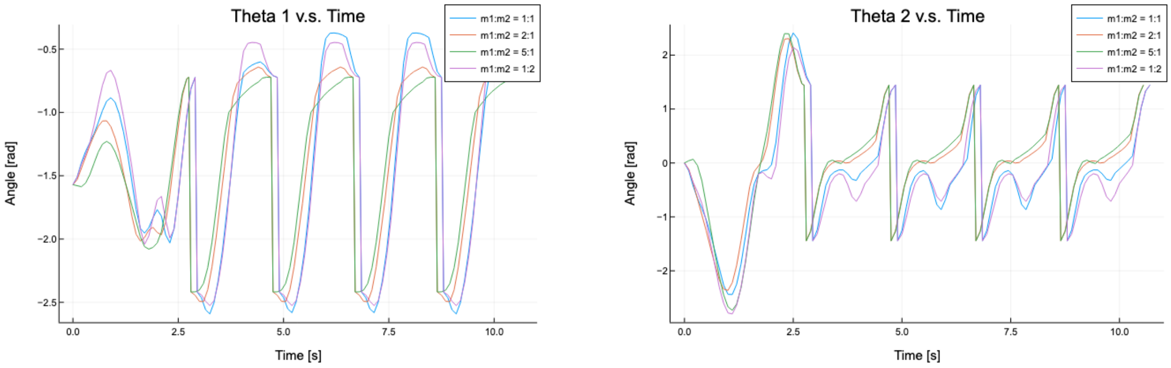

Fig. 8. Joint angle vs. Time for varying mass ratio tests

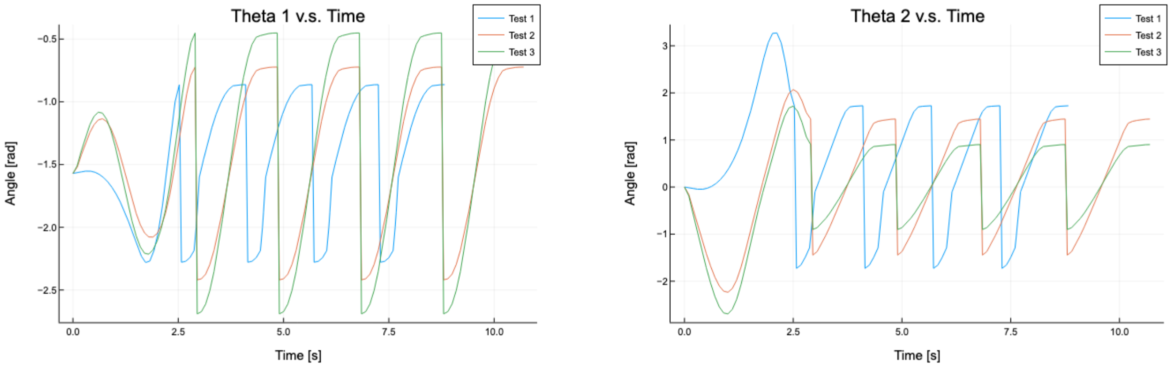

Fig. 9. Joint angle vs. Time for varying bar length tests

Since torque wasn't very heavily penalized, moving θ_2 fast to generate momentum in the system became less costly.

Looking at the tests with changing bar length, the swing up trajectory changes depending on how far away the next bar is. Looking at Fig.9, the optimal trajectory when the bars are close together appears to be swinging up into a single pendulum configuration before reaching out for the next bar. This suggests that the action of swinging out to catch the next bar is more costly as the bars get farther apart which intuitively makes sense. More torque is required to complete that extension motion as the bars get farther apart which is why the solver ends up with an extended swinging trajectory for the farther separated bars.

VI. Conclusion

After running various tests with various different cost function implementations, a simple quadratic cost function produced the most consistently feasible and meaningful results. Significant tuning of this cost function was required to get consistent feasible trajectories from the DIRCOL solver. This tuning included placing higher weights on the state variables and lower weight on input torque as well as implementing a tracking cost on the time step at each iteration of the trajectory. This free time setup allowed the solver to follow the natural pendulum dynamics resulting in very smooth reasonable trajectories.

In addition to cost function setup, mass ratio and bar separation were also crucial to the behavior of the final trajectory. Both increasing the mass of the feet and decreasing bar separation distance resulted in unusual pendulum behavior where the optimal trajectory was to swing up into a single pendulum like system before swinging out to the next bar.

Future work could extend this to hardware using some type of trajectory tracking controller to test whether these trajectories are actually physically feasible. In addition, this approach could also be extended to grab onto bars that are of different heights.

References

[1] J. Nakanishi, T. Fukuda, and D. Koditschek, “Experimental implementation of a ”target dynamics” controller on a two-link brachiating robot,” vol. 1, pp. 787–792 vol.1, 1998.

[2] T. Fukuda and F. Saito, “Motion control of a brachiation robot,” *Robotics and Autonomous Systems*, vol. 18, no. 1, pp. 83–93, 1996.

[3] C.-Y. Lin and Y.-J. Tian, “Design of transverse brachiation robot and motion control system for locomotion between ledges at different elevations,” *Sensors*, vol. 22, no. 11, 2022.

[4] J. Nakanishi, T. Fukuda, and D. E. Koditschek, “Preliminary studies of a second generation brachiation robot controller,” *Proceedings of International Conference on Robotics and Automation*, vol. 3, pp. 2050–2056 vol.3, 1997.

[5] D. Reda, H. Y. Ling, and M. van de Panne, “Learning to brachiate via simplified model imitation,” *ACM SIGGRAPH 2022 Conference Proceedings*, 2022.

[6] E. Damghani and S. A. A. Moosavian, “Brachiation robot dynamics modelling,” *2022 10th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pp. 414–420, 2022.

[7] R. Tedrake, *Underactuated Robotics*. 2023.

[8] E. Shata, P. Acharya, and K.-D. Nguyen, “Brachiating robot analysis and design,” pp. 230–235, 2019.

[9] S. Yang, Z. Gu, R. Ge, A. M. Johnson, M. Travers, and H. Choset, “Design and implementation of a three-link brachiation robot with optimal control based trajectory tracking controller,” 2019.

[10] S. Farzan, V. Azimi, A.-P. Hu, and J. Rogers, “Adaptive control of wire-borne underactuated brachiating robots using control lyapunov and barrier functions,” *IEEE Transactions on Control Systems Technology*, vol. 30, pp. 2598–2614, 2022.

[11] M. Javadi, D. Harnack, P. A. Stocco, S. Kumar, S. Vyas, D. Pizzutilo, and F. Kirchner, “Acromonk: A minimalist underactuated brachiating robot,” *IEEE Robotics and Automation Letters*, vol. 8, pp. 3637–3644, 2023.

[12] V. M. de Oliveira and W. F. Lages, “Control of a brachiation robot with a single underactuated joint using nonlinear model predictive control,” *IFAC Proceedings Volumes*, vol. 40, pp. 430–435, 2007.

[13] R. Bordalba, T. Schoels, L. Ros, J. M. Porta, and M. Diehl, “Direct collocation methods for trajectory optimization in constrained robotic systems,” *IEEE Transactions on Robotics*, vol. 39, pp. 183–202, feb 2023.