Trajectory Optimization for Continuous Contact Brachiation on Gapped Bars

Kaleb Blake and John Flynn
6.832 Underactuated Robotics
(Dated: May 10, 2022)

This paper explores the ability of a 3-link brachiation robot to swing along gapped bars. We first tackle the problem of swinging on bars that are perfectly horizontal. This is formulated as finding a limit cycle through trajectory optimization. Then, a modified model is created to solve the varying slope bars, with different rules for contact and actuation. Limit cycles for the brachiation robot are compared with those found by the double pendulum. These findings serve as a strong foothold for future exploration which we explore in the last section.

View Video Presentation of Results w Simulations: https://www.youtube.com/watch?v=3BG0AfvlN64

I. INTRODUCTION

Brachiation is a mode of transportation frequented by primates. This involves swinging with their upper limbs. With only a single contact point, the motion is underactuated and nonlinear. Besides the potential energy built up, and the limited force produced at the contact point, momentum must be generated through raising and lowering the center of momentum.

This paper is motivated by the monkey bars - a challenge humans set for themselves to imitate the expert monkeys. This can be modelled in two dimensions, dropping the dimension parallel to the bars, since the arms do not need to cross one another in brachiation. From just a simple model, one can better understanding why primates swing the ways they do, and learn new optimizations that brachiation robots could use to outperform biological swingers.

For simplicity, we look at horizontal bars most extensively. This is useful in the case of robot-centric architectural design, which will hopefully become increasingly prevalent. However, sometimes one might want to have the robot elevate or descend. Additionally, in the physical world, the "bars" are not perfectly oriented. This motivates solving the problem of swinging along arbitrarily placed bars, so long as there exists a configuration where the robot could stretch between the two.

On a personal level, we chose this project because we wanted to create a model of an existing system in the world, but one which had the potential also to surprise us with the trajectories. Trajectory optimization can have the same thrill as working on machine learning, providing you with unexpected results that are enjoyable to unpack, and can give you insights you wouldn't have arrived at had you been being more methodical about things. But mostly, when we came up with the idea of doing a monkey robot, we jumped on it, because we think that swinging is cool, understudied, and most of all, under-thought-about. How often per week do you think people actively realize swinging is a mode of transport, as opposed to walking, biking, driving, swimming, etc?

II. RELATED WORK

There has been a multitude of research on robots that can perform brachiation. In [1] a three-link robot with two arm links and one body link that can perform brachiation is researched. They look at a robot with actuation at the shoulder joints, while our project includes work on actuation of one link's hand and another link's shoulder. They used an iterative Linear Quadratic Regulator (iLQR) to solve their trajectory optimization problem and generate a viable trajectory instead of our methods of direct transcription and direct collocation. They also explore how different body mass distribution between the arms and the body and different body lengths effect the total cost of the iLQR controller. In addition they implement a trajectory tracking feedback controller that is able to perform some trajectory stabilization for the real system they built.

[2] approaches brachiation with the goal of minimizing muscle-work, therefore their models use no actuation. Their solutions have no energy cost in general. For collisions, this is achieved by modeling perfectly inelastic collisions with a coefficient of restitution of 0, which is the approach we used for collisions. However, they achieve this by specifying that the approach velocity tends to zero as contact with the hand-hold gets closer. We find an impulse that allows our perfectly inelastic collision to hold true. They also only look for solutions that are symmetric about some mid-swing configuration because asymmetric, periodic solutions with zero approach velocity have not been found. [2] explores both continued contact and ricochetal brachiation as well as various complexities of robot from a point mass model to a five-link system.

The compass gait model from MIT Underactuated Robotics Spring 2022 [3] was quite similar to our project and we drew a lot of inspiration from it. The compass gait is a three-link with two leg links and a body, which walks downhill using no actuation. Even though our systems complete different motions the implementation of contact into the manipulator equation and inelastic collision model of the three-link compass gait is quite similar to our implementation. Both our projects utilize a contact Jacobian for finding contact force of the link that is in constant contact with some element of the world (i.e.

bar, ground) and for finding the impulse generated from a perfectly inelastic collision.

III. ROBOT MODELLING AND DYNAMICS

We explored with different models, and so the work of this paper/project is not fully linear. We tried to keep it consistent as much as possible between the systems, for example keeping arm masses, body mass, costs on input and time spent, all the same. We present the different explored systems in this section.

Robot1 - Horizontal Only, with Contact

The first area of brachiation we explored was continuous brachiation of a three link robot with varying bar gap lengths. We chose to explore two different methods of actuation: wrist actuation of one link and shoulder actuation of the other link versus shoulder actuation for both links. Figures 1 and 2 show the different coordinate systems for both actuation methods. For each robot the generalized coordinates are $q = \begin{bmatrix} x & y & \theta_1 & \theta_2 \end{bmatrix}^T$. Both robots followed the following manipulator equation,

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u} + \mathbf{J}^T(\mathbf{q})\lambda_f$$

where, $\mathbf{J}(\mathbf{q})$ is the contact Jacobian of the grasp hand with respect to the middle bar and λ_f is the force vector on the grasp hand in the directions of the contact Jacobian. The manipulator equations for each robot were found using pydrake's capabilites with the urdf we constructed.

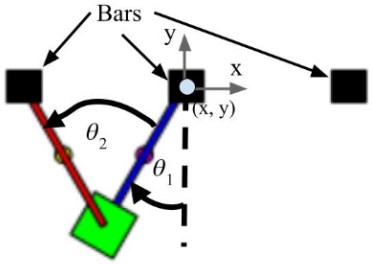

FIG. 1. Definition of general coordinates $q = \begin{bmatrix} x & y & \theta_1 & \theta_2 \end{bmatrix}^T$ for Robot1 with wrist actuation on the grasp link and shoulder actuation on the swing link.

In addition to contact of the grasp hand on the middle bar, we needed to consider the contact of the swing hand on the final bar. We decided to model the contact of the swing hand on the final bar as an impulsive collision. A collision is defined as when the distance of the swing hand to the final bar in the t_1 direction or $\phi(\mathbf{q}) \leq 0$, which can be seen in Figure 3. At this condition the following equation for impulse must be solved,

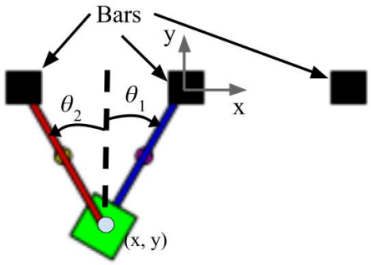

FIG. 2. Definition of general coordinates $q = \begin{bmatrix} x & y & \theta_1 & \theta_2 \end{bmatrix}^T$ for Robot1 with shoulder actuation for both links.

$$\mathbf{M}\dot{\mathbf{q}}^+ - \mathbf{M}\dot{\mathbf{q}}^- = \mathbf{J}^T(\mathbf{q})\lambda_{imp}$$

The contact Jacobian is of the swing hand with respect to the final bar. Using the contact Jacobian allows for the impulse to have tangential frictional components as well as the normal component. λ_{imp} is the total impulsive force vector from the collision in the directions of the contact Jacobian, where each component of the vector can be defined as a scalar impulse, λ_i, integrated over the time of collision from t_c^- to t_c^+. We also wanted the collision to be perfectly inelastic, meaning it had a coefficient of restitution, e, equal to 0. This means that the swing hand holds onto the bar once it collides with it, as $\dot{\phi}_c^+ = -e\dot{\phi}_c^-$ must be satisfied after the collision with $e = 0$. We enforced this with the following equation,

$$\mathbf{J}\dot{\mathbf{q}}^+ = diag(-e, 0, 0)\mathbf{J}\dot{\mathbf{q}}^-$$

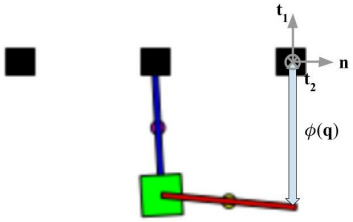

FIG. 3. The function $\phi(\mathbf{q})$ is shown which represents the distance of the swing hand to the final bar in the t_1 direction.

For the trajectory optimization, we needed to specify what the initial and final configurations would be for each type of actuated robot based on the geometry of

the system. This can be seen in 4. This figure shows how the initial and final positions could be calculated for the wrist and shoulder actuated robot. For the initial configuration we can see that $-2\theta_1(0) = \theta_2(0)$ because we made each arm link the same length. Then we can use the bar gap length to calculate $\theta_1(0) = -\arcsin(\frac{l_{b1}}{2l_a})$. For the final configuration we can use a similar equation for θ_1 inserting the new bar gap length and switching signs so that $\theta_1(t_f) = \arcsin(\frac{l_{b2}}{2l_a})$. Now θ_2 relates to θ_1 in a different way: $\theta_2(t_f) = 2\pi - \theta_1(t_f)$. Similar methods can be used to obtain the initial and final configurations of the double shoulder actuated robot. The results would be $\theta_1(0) = -\theta_2(0)$, $\theta_2(0) = \arcsin(\frac{l_{b1}}{2l_a})$, $\theta_1(t_f) = \arcsin(\frac{l_{b2}}{2l_a})$, and $\theta_2(t_f) = 2\pi - \theta_1(t_f)$.

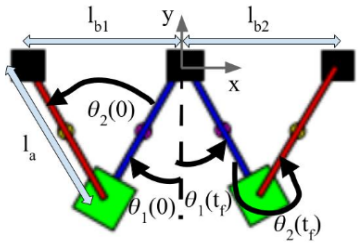

FIG. 4. The wrist and shoulder actuation robot shown in its initial and final configurations side by side. Initial and final angle configurations are shown as well as the lengths of the bar gaps and arm links. One can use the geometry of the system to calculate the initial and final angle configurations to allow the swing hand to be on the leftmost and rightmost bar, respectively.

Robot2 - Multiple Directions, no Contact

This robot aims to answer the question of "how can the robot swing between bars placed arbitrarily." Originally, it was hoped to plan multiple bars ahead, or from arbitrary starting configuration, but at the deadline for this paper the completed work was only for a limit cycle for arbitrary bar slope.

This robot is very similar to Robot1. It has a body mass which is 5 times as heavy as the masses on the arms. θ_1 and θ_2 are defined as in 5. These definitions are chosen since it makes the torques correspond to forward motion of the full body or the swinging arm respectively.

Contact with both bars is defined to halt all motion of θ_1 and θ_2. This is a reasonable assumption, since the only thing which could cause the angles to change would be a breaking in contact or a breaking of the model. This corresponds to insufficient grip strength or injuring joints from too-fast contact, respectively. Even if the stop is not instantaneous, an agent can pause until it is in position, and begin the next swing once things have stabilized. (Admittedly, this is antithesis of this class, but it is the simple version of the problem we start with. Our eventual goal is to incorporate the dynamics to our advantage and plan over multiple rungs - but this is still a tough task even in the recent literature.)

The actuators are defined to be on the angles. This is the shoulder of the swinging arm, but the hand of the hanging arm. We use this simplification because it makes coding the simulation easier, and because it seems a reasonable approximation, since the torque from the shoulder generates a full body rotation with the contact from the ceiling-bar. This system is quite analogous to the double pendulum, but with masses in different locations. (This would be one of the first things that we would fix given more time to make the actuation closer to that of a true primate)

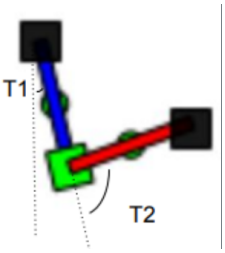

FIG. 5. Definition of θ_1 and θ_2 for Robot2

To perform trajectory optimization, we need to specify the initial and final conditions. The state is $q = [\theta_1, \theta_2, \theta_1', \theta_2']$. θ_1' and θ_2' are 0 for q_i and left as decision variables for q_f. Left to compute is θ_1 and θ_2 for q_i and q_f. Solving first for q_f (see 6).

Given r, x, z, Let capital letters be associated angles,

$$h = x^2 + z^2$$

On the big triangle,

$$R = \pi - arccos(\frac{2r^2 - h^2}{2r^2})$$

$$H = arccos(\frac{h^2}{2rh})$$

Giving us $\boxed{\theta_2 = \pi - H}$
On the small triangle on the left,

$$a = rcos(R)$$

$$b = rsin(R)$$

Then since we have the vector towards the bar, we can go a in that direction, and b in the perpendicular direction to get the coordinated of c (cx, cy). Assuming the left bar is at (0,0)

$$cx = \frac{xa + zb}{h}$$

$$cx = \frac{za - xb}{h}$$

θ_1 follows immediately since we have 3 sides of the triangle. Care is taken because the base could have to be above the horizontal.

$$\boxed{\theta_1 = \frac{\pi}{2} * u[cz] + arccos(\frac{r^2 + (cz)^2 - (cx)^2}{2r|cz|})}$$

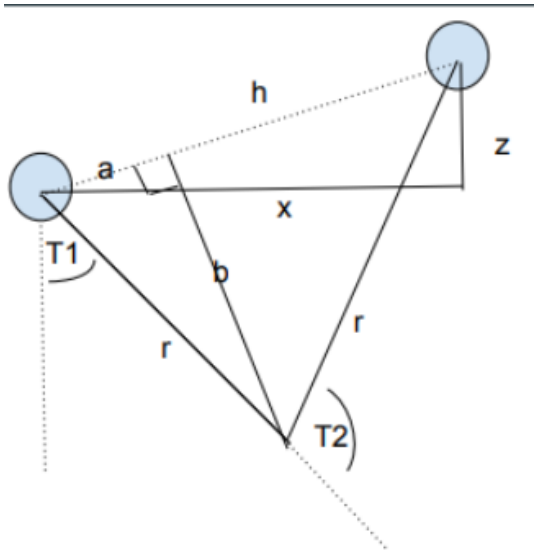

FIG. 6. Geometry used to determine initial conditions

Those were the final coordinates for θ_1 and θ_2, now for the initial positions. See 17.

θ_{2i} we get instantly because the two triangles are similar, so their supplements are equal

$$\boxed{\theta_{2i} = -\theta_{2f}}$$

We can also immediately see $theta_{1i}$ off of the straight angle.

$$\boxed{\theta_{1i} = \pi - \theta_{1f} - 2M}$$

where,

$$M = R$$

Design Parameters

We kept parameters constant between the two robots, as seen in the table below. The arm length is normalized at 1, to ease understanding of the distances cited later in the paper. The arms each account for 5% of body mass, while the torso accounts for 50%, which is how we chose the ratio for the masses. The arm mass is kept in the middle of the arm for simplicity. The torque limit

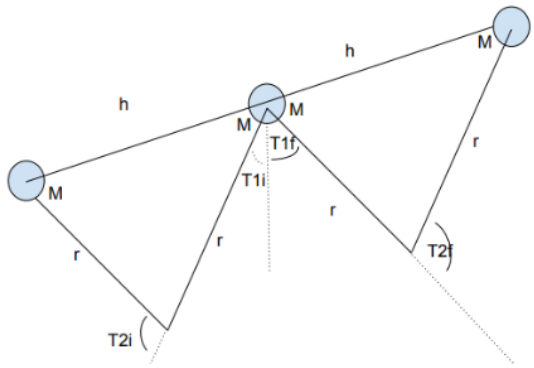

FIG. 7. Geometry used to determine final conditions

is set at 2; we got this number from saying the average person who can claim to have fitness anywhere close to a monkey can curl about 15% of their body weight reasonably comfortably. We give them an extra 5% to exert themselves. The weight is about 10N, so we get a max torque of 2N. This ends up not coming close to what is used except for swinging upwards.

arm length (r)	1
body mass	1
arm mass	.1
arm mass position	.5
torque limit	2

IV. TRAJECTORY OPTIMIZATION

Trajectory optimization is a formulation of a control problem. In it, typically an initial and final condition are specified, along with costs and constraints along the trajectory, to produce an optimization problem. This can be solved to give an "optimal trajectory" with respect to those costs and constraints. Direct transcription is used to formulate the dynamics here since we were doing a simple limit cycle. If the goal was to travel along a multi-bar path, direct shooting would need to be used. Direct collocation is used in Robot2 as the solving method, while SNOPT is used in Robot1. This was mostly to gain exposure to different ways of solving the problem. Below, the specific transcription of the problem is outlined in more detail.

For both, we chose to constrain the total time and the actuation roughly equally to get a good generic solution, rather than having only time in the cost function which would result in speed but high effort, or vice versa which would give us a slow energy conserving solution.

Robot1 - Horizontal Only, with Contact

The trajectory optimization for the robots with contact modeled for bars of different horizontal gap

lengths was done using the direct transcription method and solved using the SNOPT solver. The following decision variables are used for the optimization problem: $\mathbf{q}[t]$, $\dot{\mathbf{q}}[t]$, $\ddot{\mathbf{q}}[t]$, $\mathbf{u}[t]$, $\lambda_f[t]$, λ_{imp}, $h[t]$, and $\dot{\mathbf{q}}^+$ for all t from 0 to t_f. $h[t]$ is a variable for the time between consecutive time steps t and $t+1$. $\dot{\mathbf{q}}^+$ is the time derivative of the generalized coordinates after the swing hand collision with the final bar. The cost we minimize is $\sum_{t=0}^{t=t_f} \mathbf{u}[t] \cdot \mathbf{u}[t]$. To ensure the problem gives a valid solution we impose several constraints. We enforce the manipulator equation, impulsive and inelastic collision, and initial and final robot configuration constraints discussed in the previous section. We also used the implicit Euler method to link the configurations, velocities, and accelerations.

$$\mathbf{q}[t+1] = \mathbf{q}[t] + h[t]\dot{\mathbf{q}}[t+1]$$

$$\dot{\mathbf{q}}[t+1] = \dot{\mathbf{q}}[t] + h[t]\ddot{\mathbf{q}}[t+1]$$

We needed to enforce periodicity between of the absolute angular velocity of each link, such that the angular velocity of the swing arm just after grasping the final bar was equal to that of the grasp arm at $t = 0$. This is because the swing arm becomes the grasp arm after grasping the final bar. The symmetric opposite should also be true, meaning the angular velocity of the grasp arm just after the swing arm completes its swing, should be equal to the angular velocity of the swing arm at $t = 0$. This constraint ends up having different forms for the different actuation methods of Robot1. For the wrist and shoulder actuated robot, we can see from 8 that if the blue link was the swinging link, its value of θ_1, θ_{1b}, would be equal to $\theta_1 r(t) + \theta_2 r(t)$, which are the generalized coordinate angles when the blue link is grasping (red link is swinging). Therefore to make the angular velocity of the swing arm just after grasping the final bar equal to that of the grasp arm at $t = 0$ the following must be true,

$$\dot{\theta_1}(0) = \dot{\theta_1}(t_c^+) + \dot{\theta_2}(t_c^+)$$

or, in terms of the zero-indexed decision variable

$$\dot{\mathbf{q}}[t][2] = \dot{\mathbf{q}}^+[2] + \dot{\mathbf{q}}^+[3]$$

The symmetric opposite can be enforced in a similar way with,

$$\dot{\theta_2}(0) = -\dot{\theta_2}(t_c^+)$$

For the double shoulder actuated robot we can use similar methods to find the following equations to enforce periodicity between the angular velocities of links.

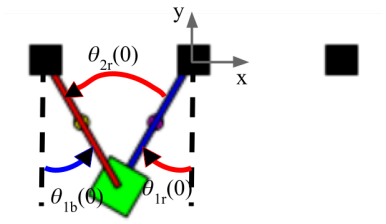

FIG. 8. A visualization of the blue grasp link in the generalized coordinates at the start of the swing cycle $(\theta_{1r}, \theta_{2r})$ versus the generalized coordinates for blue link being the swing link (imagine the red link was blue, and focus on θ_{1b}). This helps to show how one can calculate the equation to enforce the angular velocity of the grasp link at $t = 0$ (blue link) is equal to the swing link angular velocity at $t = t_c^+$ (also the blue link).

$$\dot{\theta_1}(0) = \dot{\theta_2}(t_c^+)$$

$$\dot{\theta_2}(0) = \dot{\theta_1}(t_c^+)$$

Some additional constraints we had to include were fixing the grasp hand on the middle bar, and keeping the swing hand from colliding with the middle bar or the final bar mid-swing by keeping it under $y = 0$ from $t = 1$ to $t = t_f - 1$. All these constraints and costs were fed into the SNOPT solver.

Robot2 - Multiple Directions, no Contact

This trajectory optimization was done using *direct collocation* on pyDrake. A random reachable bar (distance < 2 from start) is generated, and a few of these are placed on the urdf. For this direct collocation, 30 time steps were specified, with a min time of 0.02 and max time of 0.05 for each step. This means that the trajectory could take anywhere from .6 - 1.5 seconds.

Constraints were then added. The time was constrained to have equal intervals so the optimization would not take forever but still give us the desired effect of variable total time. The initial and final conditions calculated in the previous section were imposed. There was also a torque constraint added to all knot points, keeping the input torque to reasonable exertion levels, 2. The last constraint was for θ_1 never to exceed the angle between the bars. In other words, this meant that the body never breaks the plane between the two bars. This is meant to prevent unrealistic optimizations where the body is hoisted above the bars.

There were only two costs. First, a cost on the total time taken. Second, a cost equal to the magnitude of the input.

This program can then be passed to drake's solver.

V. RESULTS

Robot1 - Horizontal Only, with Contact

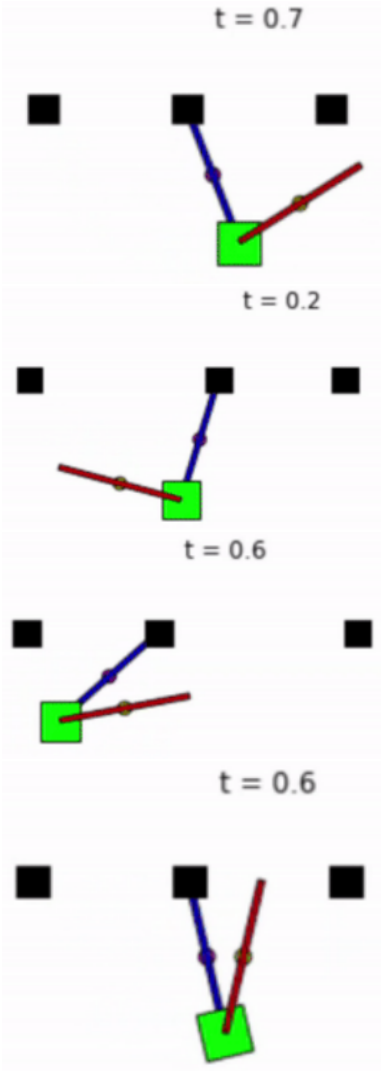

FIG. 9. This figure shows screenshots of various double shoulder actuation Robot1 simulations. The solutions for the wrist and shoulder actuated Robot1 are similar. From top to bottom we see the solution for a 1m gap to a 1m gap, 1.5m gap to a 1m gap, 1m gap to a 1.5m gap, and lastly an upside-down walking solution. The upside down-walking doesn't follow the swing trajectory optimization discussed above, but looks more like an upside-down walking motion where the swing link stays at $y = 0$ for the duration of the motion.

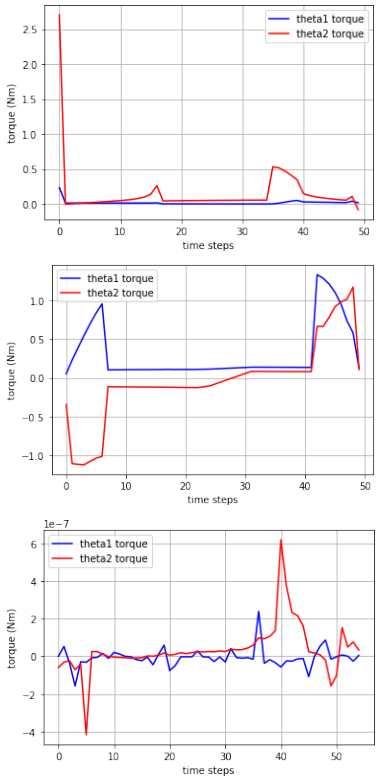

FIG. 10. This figure shows the trajectory optimization torque values – $\tau_{\theta_1} = \mathbf{u}[t][0], \tau_{\theta_2} = \mathbf{u}[t][1]$ – for the wrist and shoulder actuation, upside down walking, and double shoulder actuation from top to bottom, respectively.

Robot2 - Multiple Directions, no Contact

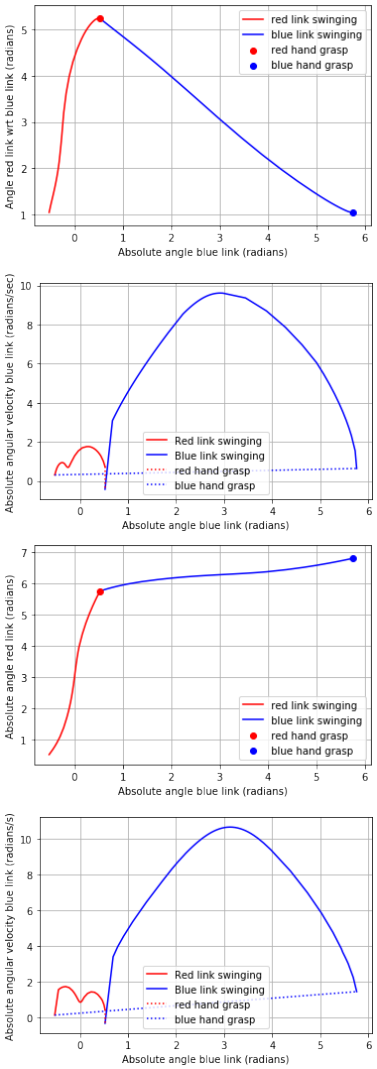

FIG. 11. This figure shows the two types of limit cycles for each actuation method of Robot1: limit cycle of θ_2 versus θ_1 and of $\dot{\theta}_1$ versus θ_1. The first pair of limit cycles are from the wrist and shoulder actuation method, while the second two are from the double shoulder actuation method. These limit cycles were created for 1m to 1m bar gaps because those trajectories would form more continuous cycles.

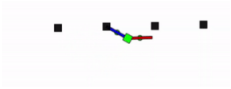

FIG. 12. Representation of Robot2 going straight

VI. DISCUSSION

Despite the early stage of this research, there are already several interesting takeaways. A few things can be observed directly from the simulations. Comparing with "upside-down walking", we see that brachiation is fundamentally different from walking, despite the similarities. Also, it is seen when the bar to swing to is relatively far

FIG. 13. Representation of Robot2 going upwards

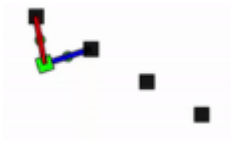

FIG. 14. Representation of Robot2 going downwards

away, the free arm approaches from the left of the bar, while when it is close, the arm overshoots the bar and comes back. This behavior, typically going unnoticed, is how most people probably would deal with this bar setup.

Another thing to note is the difference in costs with the different models in the horizontal, equal spacing case. Walking had a cost of 24.06, the hand, shoulder double-pendulum-inspired model had cost of 8.67, while the double-shoulder had a cost approximately zero. It is amazing how this swing is approximately passive, an excellent means of transportation, and this lines up with the results of [2].

The limit cycles on the generalized coordinate angles obtained from Robot1 do not form loops because at least one of the angles analyzed continuously increases

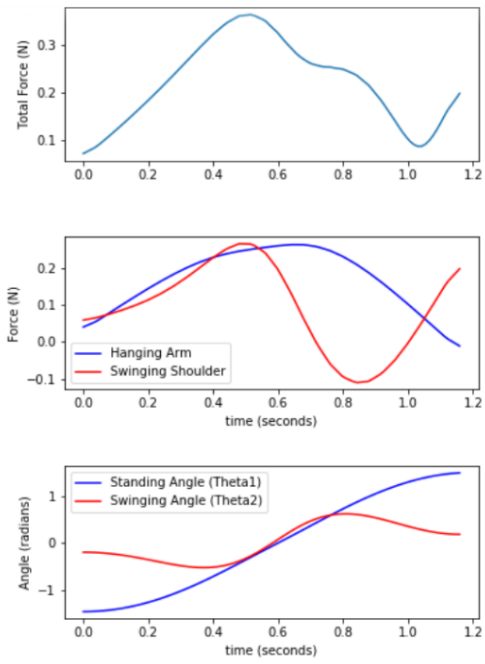

FIG. 15. (Top): Torque/Position information for **Flat** trajectory

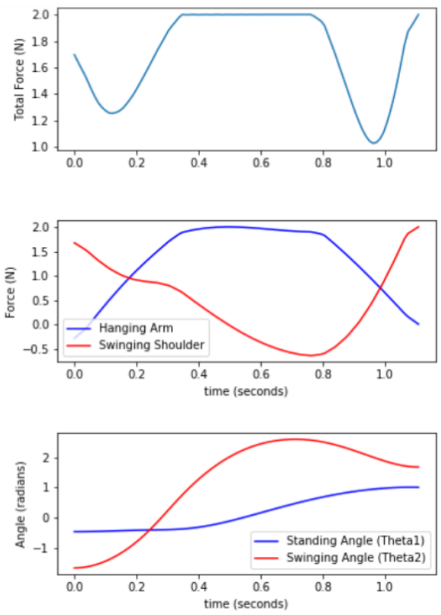

FIG. 16. (Top): Torque/Position information for **Upwards** trajectory

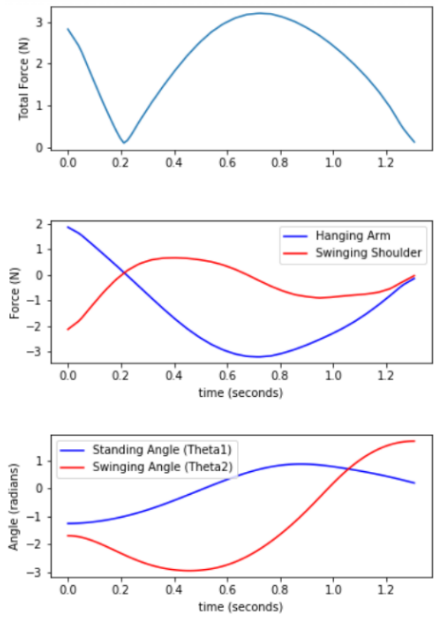

FIG. 17. (Top): Torque/Position information for **Downwards** trajectory. Evidently, the torque limit was not enforced on this trajectory which is a bad error.

throughout the swinging period of the red link and the swinging period of the blue link. For instance, in the wrist and shoulder actuation θ_1 or the absolute angle of the blue link increases for a small amount when the red link is swinging and the blue link is grasping. But once the blue link is swinging θ_1 increases greatly. On the other hand, θ_2 or the angle of the red link with respect to the blue link oscillates between a minimum and maximum value. So for the limit cycles on the generalized coordinate angles there is always one jump. While looking at the limit cycles for $\dot{\theta}_1$ versus θ_1, there are two jumps.

It is important to keep in mind that Robot2 is essentially a double pendulum actuator with primate mass placement. Given the contact model, we do believe this can still tell us some interesting things about swinging and primate motion.

First, in the simulations, we notice that the swinging upwards is analogous to the swinging far case, where the free arm approaches from the left. Likewise, swinging downwards is analogous to swinging near, where the arm approaches from the right. Imagine climbing upwards, most likely you would climb forwards, whereas climbing downwards, it might be more stable to come in from the back, because of gravity is working against your attempt to swing up your arm too early, so you wait and use the tension of your hanging arm.

For torque on the hanging arm, there is a similar pattern with going straight and up. There is a ramping up and down of the torque. For the upward swing, this ramp is much stronger (in fact it reaches the torque limit), as would be expected to generate the force to swing up. The hanging arm on the downward swing does the opposite, acting as a stabilizer against the gravity. For all, the torque goes to 0 at the final step, meaning the swinging shoulder is used to guide the robot onto the bar. For torque on the swinging shoulder, on the flat surface, there is a gentle forward then back torque, just used to stabilize, most of the work is done by gravity. For the upward climb, there is a strong swing to move the arm in front. For the downward climb, the arm is torqued negatively and brought together with the other arm, so the body can be swung together, and then it is positively torqued all the way around (heavily helped by gravity) to arrive on the right side of the bar. This is an interesting trajectory and probably not one that would be taken in reality, but in the extreme straight down case you can imagine how one would not just swing the same way as horizontal, and might hedge their way down like this trajectory.

Lastly, in comparison to simulations of the normal pendulum, effort expended (torque applied) is significantly lower for straight and downward swings, however it is more for the upward swings. Why did we evolve to have our weight distributed this way? Perhaps we are meant to swing more straight and down than upwards. Climbing may be the preferred method to gain gravitational potential energy.

At a high level, though, these trajectories follow the main pattern one would expect. The straight swing is very reminiscent of monkey brachiation. The upward swing maxes out the torque limit, as expected it is more difficult to swing upwards. Also, the robot swings hand in from the left on the way up, while from the right on the way down.

** trajectories discussed can be found in the video

linked at the start of the paper **

VII. FUTURE WORK

We had many ambitious plans coming in to the final project, but as warned learning how to use drake and make a proper simulation were harder than expected. Through this struggle though, the structure of the package is coming together, how MathematicalPrograms, MultibodySimulations, Visualizations, URDFs, all fit together. So there are many ideas for where to go next, some new, some we've had from the beginning. We'll conclude the paper by outlining a few of these.

1) Expand to a 5-link system, 2 links per arm to make a more realistic arm. Can try different actuations (e.g. actuation at shoulders, passive elbows)

2) Create controller and simulate using uncertainty in the dynamics modelling. Introduce some ϵ into the dynamics, and show that the controller can stabilize the whole region around the trajectory. Potentially use closed form analysis for this. (Lyapunov on linearization around trajectory? Likely no but maybe)

3) Try using iLQR instead of trajectory optimization to see if one can obtain better results or more robustness

4) Create a real (physical) system and use trajectory stabilization to ensure success

5) Optimize over multiple bars at once. Many modern papers are still working just on single bar planning, so the multi-bar is a difficult problem. It would be really cool to see how the optimizer uses the dynamics to achieve better performance on two rungs than on each individually, given an appropriate contact model.

6) Add new constraints (e.g. injured arm, lower torque limit)

7) Test if this far = up , close = down relationship holds in more scenarios. Look for more patterns in torque timing.

VIII. CONTRIBUTIONS BY EACH MEMBER

John: Worked on the problem of arbitrary bar placement trajectories.

Kaleb: Worked on the horizontal limit cycle problem.

Both: Planning/focusing the project, determining equations of motion, learning how to write mathematical programs and simulations in drake, making slideshow, writing paper.

ACKNOWLEDGMENTS

We acknowledge Russ Tedrake and Lujie Yang for giving us advice and guidance on our project. We would also like to thank the pioneers that came before us in the field and creators of the pyDrake software.

[1] "Design and Implementation of a Three-Link Brachiation Robot with Optimal Control Based Trajectory Tracking Controller," (2019).

[2] "A five-link 2D brachiating ape model with life-like zero-energy-cost motions," (2005).

[3] "Underactuated Robotics Webpage," (2022).