Walking and Jumping of a Bipedal Robot

Joseph Bell
University of California, San Diego
Department of Electrical and Computer Engineering
jjbell@eng.ucsd.edu

Saurabh Mirani
University of California, San Diego
Department of Electrical and Computer Engineering
smirani@eng.ucsd.edu

Shiladitya Biswas
University of California, San Diego
Department of Electrical and Computer Engineering
s1biswas@eng.ucsd.edu

James Salem
University of California, San Diego
Department of Electrical and Computer Engineering
jgsalem@eng.ucsd.edu

***Abstract*—This paper explores legged robotic control - specifically, walking and jumping. A bipedal robot, supported by a hind set of wheels, was developed to walk a predetermined distance and jump up a step. The results were satisfactory as the robot was able to successfully walk and jump.**

***Index Terms*—bio-inspiration, bipedal, robot, locomotion**

I. Introduction

Bio-inspired robotics is concerned with the development of robots through the lens of nature. One major characteristic of the natural world that has been used as inspiration for robotic development is locomotion. Organisms traverse the natural world in a variety of ways - with legged locomotion being a popular method used by organisms. An abundance of time and money has been put toward researching legged locomotion in robotics as the applications are abundant. Although Spot, developed by Boston Dynamics, is arguably the most popular legged robot in mainstream media, the inspiration for the robot developed for this paper came from the Stanford Doggo [1]. This paper will next introduce the project goal, followed by the technical approach taken to meet the goal, and finally results will be displayed and discussed.

Fig. 1. The inspiration for our robot: Stanford Doggo

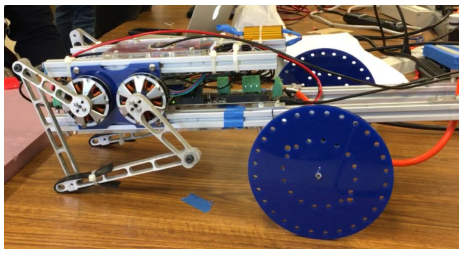


Fig. 2. Side view of our robot

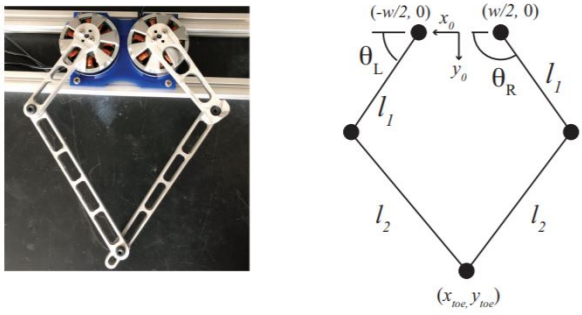


Fig. 3. Leg configuration

II. Project Goal

The goal of this paper is to explore legged locomotion in a bipedal robot. The method of locomotion chosen by our group was a combined sequence of walking and jumping. The goal was to develop a robot to walk and jump up a step of approximately two inches.

III. Technical Approach

A. Slipping

Locomotion of any kind typically requires pushing against something, taking advantage of Newton's third law of motion. In the case of both bipedal walking and jumping, the foot pushes back along the ground resulting in the force of static

friction propelling it forward. The maximum force achieved by static friction is described by the following equation

$$F_{max} = \mu_s N$$

From the equation above, it is clear that to increase the maximum force of static friction one could either increase the coefficient of static friction, μ_s, or increase the normal force, N. To address the issue of slipping, we decided to increase the coefficient of static friction by adding rubber strips to the bottom of the robot's feet. Increasing the surface area of the feet would also help to increase the coefficient of static friction, however, this would be challenging with our 5 bar leg setup. Another way to increase the maximum force of static friction is to increase the normal force. This option will be discussed further in the following section.

B. Weight Distribution

As shown in Fig. 2, our robot has two legs in the front, a set of wheels near the middle, and a tail that extends back. With this setup, the wheels act as a fulcrum and the robot body acts as a lever system. For the robot to stand or walk, the torques acting on the lever must be balanced. Using this simple lever system model for our robot, as shown in Fig 4. , we can estimate the force exerted by the legs to keep the robot standing i.e. keep the lever system balanced.

$$\sum \tau = (m_{front}g)r_{front} - (m_{tail}g)r_{back} - F_{legs}r_{legs} = 0$$

To keep this system balanced, the larger the mass of the tail, m_{tail}, the smaller the force exerted by the legs, F_{legs}, needs to be in order for the robot to stand. This affects the ability of our robot to both walk and jump. For the walking case, the force required to be exerted by the legs to stand affects whether the robot will slip or not. The magnitude of the force, F_{legs}, is equal to the magnitude of the normal force, N. Therefore, if F_{legs} is too small, the maximum force of static friction will be small and the robot will experience slipping and will have trouble propelling itself forward. So in order to reduce slipping, the robot requires a balance of the tail mass, m_{tail}, and the mass distance from the fulcrum, r_{back}, so that the force exerted by the legs is not too small.

On the other hand, the smaller the force is required to be exerted by the legs, the easier it is to jump. If most of the robot's front weight is being balanced by the weight in the back, the legs have a significantly reduced weight to launch upward during a jump.

Since it is easier to walk if F_{legs} is larger and easier to jump if F_{legs} is smaller, in order for a robot to be able to both walk and jump, a compromise must be achieved so that the robot can walk without slipping but also not have to launch too much weight into the air during a jump.

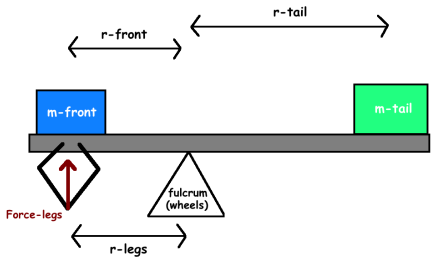


Fig. 4. Robot modelled as lever system

C. Jumping

1) Starting Position: In order to find a suitable starting position for our jump sequence, we can compute the motor torques required to exert a desired force from various starting positions. We can calculate the torque as follows:

$$\tau = J^T F$$

where J^T is the transpose of the Jacobian at a specific starting position and F is the desired force. Current can be calculated from torque from the following equation:

$$I = \frac{\tau}{K_t}$$

where I is the current in amperes and K_T is the motor torque constant. The best starting position would be the one that requires the least torque and therefore the least current.

For this simulation, we used a desired force of $F = [-25, 25]^T$ which corresponds to a force into the ground at a 45° angle. This force would allow the robot to jump both upward and forward. We performed the simulation for the following 10 configurations which are specified by the robot foot position in x, y coordinates in meters:

1) (-0.08, 0.08)
2) (-0.08, 0.15)
3) (-0.05, 0.08)
4) (-.05, 0.15)
5) (0, 0.08)
6) (0, 0.15)
7) (0.05, 0.08)
8) (0.05, 0.15)
9) (0.08, 0.08),
10) (0.15)

The results of this simulation are shown in Fig. 5 and 6. It is clear that configuration 7 requires the least current from motor 0. While configuration 7 does not require the least current from motor 1, configuration 7 is the configuration that requires the most reasonable currents from both motors. Any other configuration requires a high current from at least one of the motors. Therefore has been selected as the best starting position out of the tested set. Configuration 7 is shown in Fig. 7.

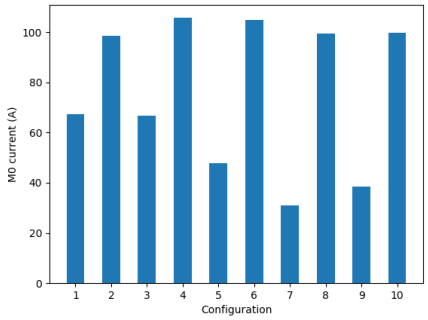


Fig. 5. Current required to exert force $F = [-25, 25]^T$ by motor 0

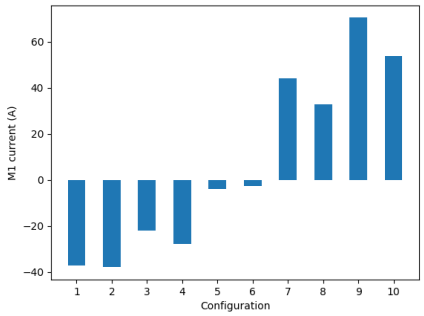


Fig. 6. Current required to exert force $F = [-25, 25]^T$ by motor 1

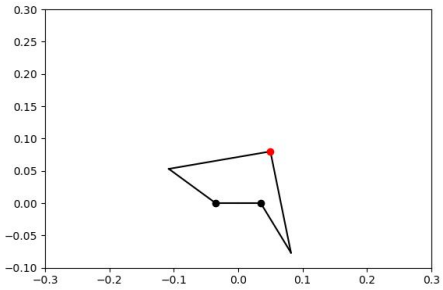


Fig. 7. Configuration 7 - robot facing to the right

2) Extension: The approach to extension was a simple one. Since jumping is the result of accelerating one's center of mass to oppose and exceed the acceleration due to gravity, we simply programmed the robot to extend its legs as fast as possible. Not only does the extension need to be quick, but it needs to be at an angle such that the generated force is sufficient in the y-direction to overcome gravity and sufficient in the x-direction to propel the robot forward. From the position illustrated in Fig. 7, the robot extends its legs down and back (up and to the left regarding the orientation of Fig. 7) in order to create reactive forces that propel the robot up and over the step (i.e. in order to replicate the force vector used in the analysis presented in the previous jumping section). Attempts were made to use feed-forward current, however, more often than not the current limit of the motors was hit and the motors would shut down. Even when the motors did not shut down, notable improvements were not noticed.

D. Walking

1) Passive vs Active Stability: A chair is passively stable, because it does not need any control or adjustment to stay upright. A standing human is actively stable because your body requires constant position control to stay standing.

When our robot is standing on two legs and two wheels, it is actively stable, that is it requires the motors to be powered to maintain the standing phase.

2) Dynamic vs static stability:

Static stability:

- Bodyweight supported within the support polygon of the legs
- Can stop moving and remain stable
- Safe, slow, but potentially inefficient

Dynamic Stability:

- Robot will fall if not continuously moving
- Less than three legs can be on the ground at a time
- Fast, efficient, but demanding for control and actuation

3) Gait Generation:

1) **Generate Parabola:** in xy plane using 3 pivot points, they are, start point of the stance phase, end point of the stance phase and mid point of the stance phase.
2) **Duty factor:** which is defined as:

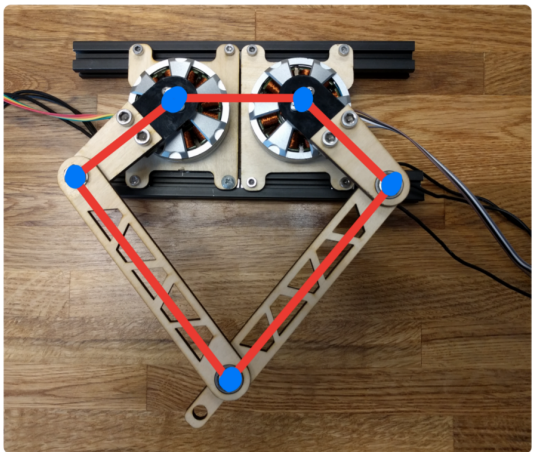
$$duty\ factor(\beta) = \frac{\text{Stance time}}{\text{Stride period}}$$

3) **Inverse Kinematics:** For all the generated points in xy plane, (θ_L, θ_R) is obtained which can be provide to the motors to reach a certain configuration.

β is set to 0.75 as we are working in static stability. This ensures that at any given point of time, the robot has at least three contact points with the ground, two wheels and 1 leg. The height of the parabola is small, that is gait trajectory is flat, having large radius of curvature. If the height is kept large then it will lead to instability, because the robot has to lift the leg higher. This results in imbalance. To avoid this, flat trajectories are used.

In Fig. 8, the red parabolic line shows the trajectory in xy plane.

Fig. 9 shows the comparison between two trajectories. The blue is flatter as compared to red one and results in proper balance of the robot.



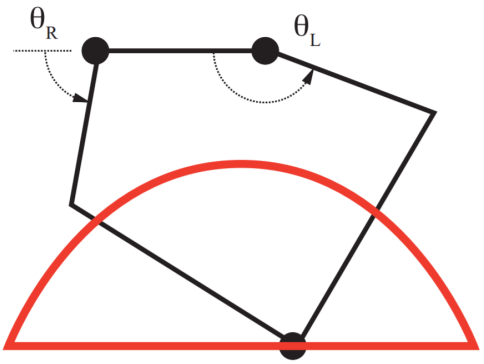


Fig. 8. Parabolic Gait Generation

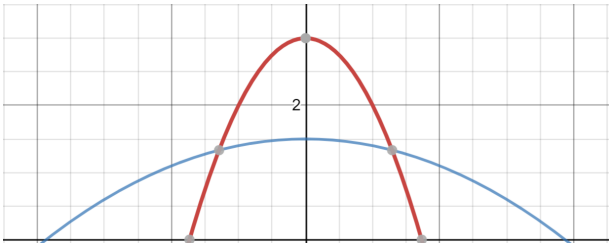


Fig. 9. Trajectory Comparison

IV. Results and Discussion

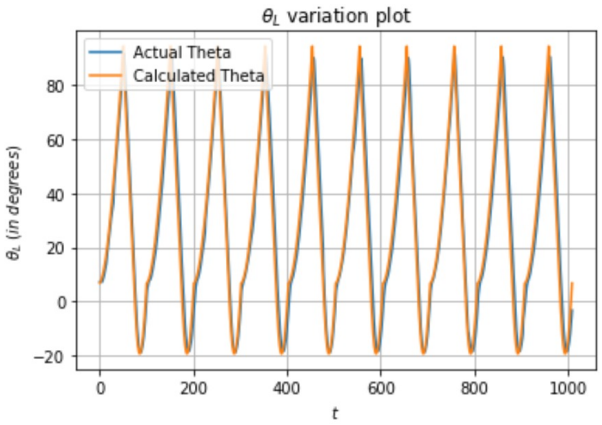


Fig. 10. θ_L variation

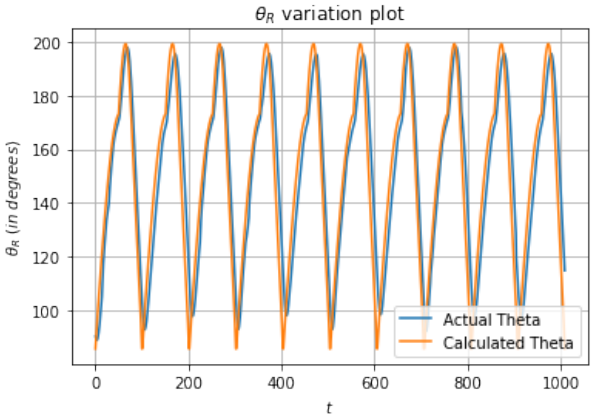


Fig. 11. θ_R variation

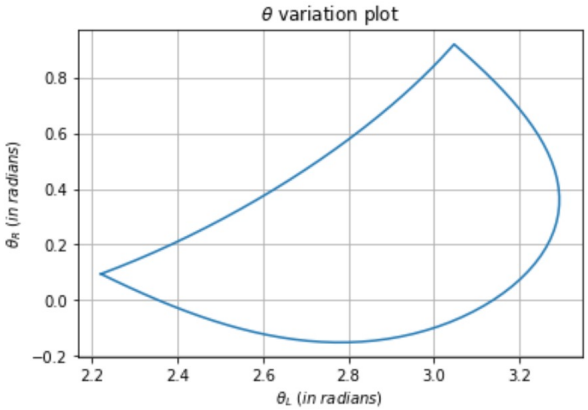


Fig. 12. θ_L vs θ_R

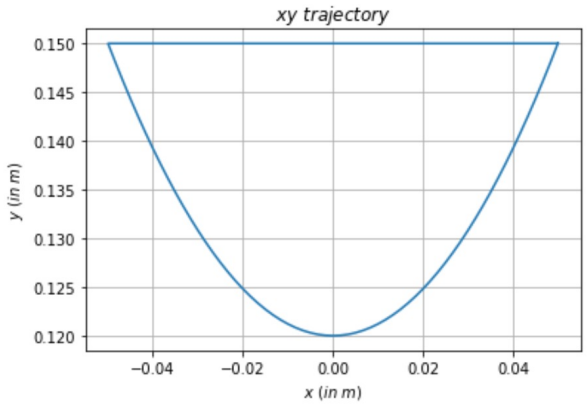


Fig. 13. xy trajectory of gait

Fig. 10 and 11 show how accurately the motors are able to reach a particular configuration. Since, the calculated and experimental values almost overlap, it can be said that the implementation is pretty accurate.

Fig. 12 is the variation of $theta_L$ vs $theta_R$ for our walking gait cycle. This cycle data is given in a loop continuously to get the desired walking of the robot.

Fig. 13 is the parabolic trajectory in xy. As it can be seen, the y variation is just 3cm, to ensure a flat trajectory for proper balancing. If the y variation is increased to about 6cm, the robot is no longer able to balance itself, it starts wobbling and hence the variation is kept low.

Here is a link that takes you to a download for the final demonstration of our robot. As one can see, the robot cleanly walks forward and jumps up the step which allows us to conclude the project was a success. One of the major complications of the project was developing a counter weight approach that worked for both walking and jumping. As the motors were not strong enough to get the weight of the robot off the ground, a counter weight was necessary to relieve the amount of force required by the motors. Initial testing without the counter weight failed to see the robot leave the ground, and along with that, the motor temperature was getting too high. We had initially used a heavy counter weight (relative to the final robot configuration) in order to generate enough counter torque about the axis of rotation as the lever arm was not that long. However, this counterweight ended up being too substantial for the robot to walk with. The hardware was readjusted such that the lever arm was longer, but the counter weight was lighter. This effectively creates the same counter torque with less weight. However, this extended lever arm means that the maximum angle of rotation about the rotation axis is shortened, and therefore the maximum jump height is decreased.

The jump height is proportional to the angle of rotation of the lever arm about the rotation axis (i.e. a greater jump height increases the angle of rotation about the rotation axis). The analysis is given below where X is the height of the robot above the ground and the lever arms are scaled according to this height. The distance of the front of the robot to the rotation axis is constant between the two configurations and is given the value 2X.

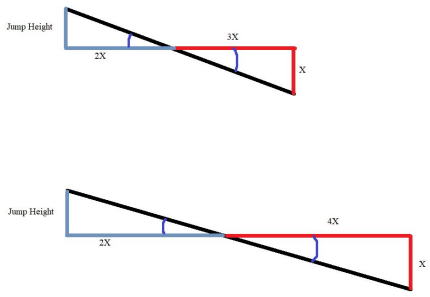


Fig. 14. Counterweight diagram: short(top), long(bottom)

$$J = 2X \cdot \tan\theta$$

where J is the jump height and

$$\theta_{short} = \arctan\frac{X}{3X}$$

for the shorter lever arm configuration, and

$$\theta_{long} = \arctan\frac{X}{4X}$$

for the longer lever arm configuration.

Solving for θ_{short} one obtains approximately 18.4 degrees. Therefore, the maximum jump height is approximately $0.66 \cdot X$. Whereas, solving for θ_{long} one obtains approximately 14 degrees. Therefore, the maximum jump height is approximately $0.5 \cdot X$. Since the extended lever arm shortened the maximum angle of rotation, the robot could not jump as high due to the lever arm striking the ground. Therefore, the jump height had to be lowered to accommodate the adjusted hardware.

Additionally, although the leg extension approach worked, it was quite simple. We had hoped to explore the performance between the simple extension method (start at point A and go to point B) versus a planned trajectory that resembled a more cyclical motion, but did not have enough time due to time lost adjusting the hardware. It's possible that a planned jumping trajectory with a current feed forward would be more elegant, but in regard to jump height, the simple approach easily hit the jump height limit. The more complicated jumping approach would not have been beneficial regarding jump height, but we hypothesize that it would be much more applicable to a four-legged running robot hurdling over objects in order to maintain the cyclical motion of the legs for a smooth transition from jumping to running.

REFERENCES

[1] Nathan Kau, Aaron Schultz, Natalie Ferrante, and Patrick Slade. Stanford doggo: An open-source, quasi-direct-drive quadruped. *CoRR*, abs/1905.04254, 2019.