Hopp3r: a Planar 3 Degree of Freedom Hopping Robot

June 8, 2018

Contents

Chapter 1

Project Definition

Chapter 2

Team Members

Chapter 3

System Diagrams

Chapter 4

Engineering Analyses

4.1 Kinematics

4.1.1 Motivation and mobility

Hopp3r is a closed-chain robot designed to study legged locomotion. Such a robot is “stronger” than a serial robot because closed kinematic chains allow multiple actuators to contribute to the motion of an end-effector. Additionally, this closed-chain legged robot has more mass concentrated at its top, compared to an open-chain robot, which is advantageous for legged locomotion for two reasons:

- Less mass near the foot increases agility and reduces actuation requirements at the hip.
- The robot behaves like an inverted pendulum during stance, so concentrating the mass at the top of the pendulum increases stability.

The remainder of this chapter derives a kinematic model of a particular planar 3-degree-of-freedom (DOF) closed-chain robot. Having 3 DOF is desirable because a planar legged robot with 3 DOF can fully control its position and orientation during stance and flight. [1]

These analyses closely follow [1], especially Chapters 6 (”Inverse Kinematics”) and 7 (”Kinematics of Closed Chains”).

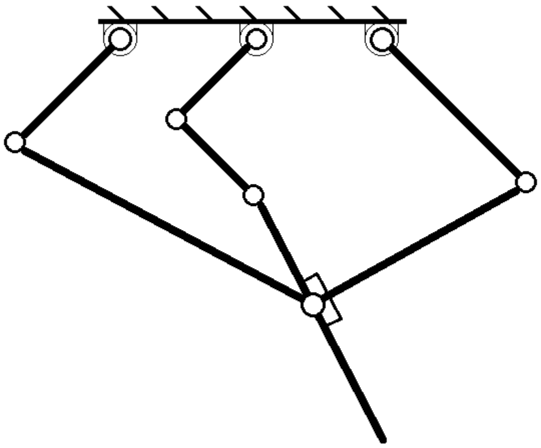

Figure 4.1: Schematic diagram of the planar robot.

We now provide a verification that the robot shown in Figure 4.1 has 3 degrees of freedom, using Grübler’s formula:

[1] The interface between the foot and ground can be viewed as an unactuated joint, inviting questions about underactuation and controllability for this particular robot.

$$\text{DOF} = m(N - 1 - J) + \sum_{i=1}^{J} f_i, \tag{4.1}$$

where $m = 3$ for planar mechanisms, N is the number of links (including the base), J is the number of joints, and f_i is the number of freedoms at the i^{th} joint. From Figure 4.1, $N = 8$, $J = 9$, and $f_i = 1$ for all i. Note that the lowest joint in the figure is really two joints; one joint connects the left coupler to the middle link, and the other connects the right coupler to the middle link. Thus, according to Grübler's formula, the robot has 3 degrees of freedom.

4.1.2 Forward kinematics

Closed-chain linkages can be decomposed into groups of open-chain linkages (hereafter referred to as ”open sub-chains”) whose motions are constrained by loop-closure equations. Our robot comprises three open sub-chains, shown in Figure 4.2: the θ-chain is on the left, the ϕ-chain is in the middle, and the ψ-chain is on the right. We will first examine the forward kinematics for each open sub-chain and then develop loop-closure equations.

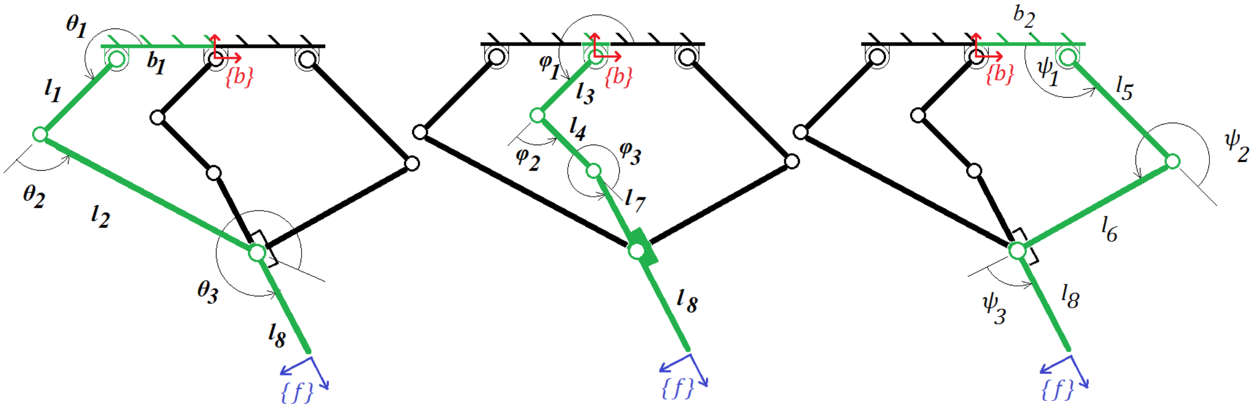

Figure 4.2: The planar robot comprises three open sub-chains, referred to as the θ-, ϕ-, and ψ-chains, from left to right.

Open sub-chain forward kinematics

The forward kinematics of each open sub-chain relate the foot frame $\{f\}$ to the base frame $\{b\}$. The forward kinematics of the θ-chain are

$$\begin{bmatrix} x_f \\ y_f \\ \measuredangle_f \end{bmatrix} = \begin{bmatrix} -b_1 + l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_8 \cos(\theta_1 + \theta_2 + \theta_3) \\ l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_8 \sin(\theta_1 + \theta_2 + \theta_3) \\ \theta_1 + \theta_2 + \theta_3 \end{bmatrix}. \tag{4.2}$$

The forward kinematics of the ϕ-chain are

$$\begin{bmatrix} x_f \\ y_f \\ \measuredangle_f \end{bmatrix} = \begin{bmatrix} l_3 \cos\phi_1 + l_4 \cos(\phi_1 + \phi_2) + (l_7 + l_8) \cos(\phi_1 + \phi_2 + \phi_3) \\ l_3 \sin\phi_1 + l_4 \sin(\phi_1 + \phi_2) + (l_7 + l_8) \sin(\phi_1 + \phi_2 + \phi_3) \\ \phi_1 + \phi_2 + \phi_3 \end{bmatrix}. \tag{4.3}$$

Lastly, the forward kinematics of the ψ-chain are

$$\begin{bmatrix} x_f \\ y_f \\ \measuredangle_f \end{bmatrix} = \begin{bmatrix} b_2 + l_5 \cos\psi_1 + l_6 \cos(\psi_1 + \psi_2) + l_8 \cos(\psi_1 + \psi_2 + \psi_3) \\ l_5 \sin\psi_1 + l_6 \sin(\psi_1 + \psi_2) + l_8 \sin(\psi_1 + \psi_2 + \psi_3) \\ \psi_1 + \psi_2 + \psi_3 \end{bmatrix}. \tag{4.4}$$

Thus, if we know the joint positions of any of the three open sub-chains, we can calculate the end-effector pose. Although this formulation is straightforward, it is not particularly useful on its own, because both the actuated and unactuated joint positions in that chain must be known in order to calculate the end-effector pose. It would be more useful to have a mapping from the positions of only the actuated joints to the end-effector pose.

Geometric forward kinematics

One way to determine the foot pose $(x_f, y_f, \measuredangle_f)$ from the actuated joint positions $q_a = (\theta_1, \phi_1, \psi_1)$ is to realize that each rotating link sweeps out a circle, so any two open sub-chains meet at a point that is the intersection of two circles (in the case of a spatial robot, these points are the intersections of spheres instead of circles). As shown in Figure 4.3, there are two intersections to consider:

- (x_A, y_A): the intersection of the θ-chain and the ψ-chain, and
- (x_{uA}, y_{uA}): the intersection of the ϕ-chain and the most distal link

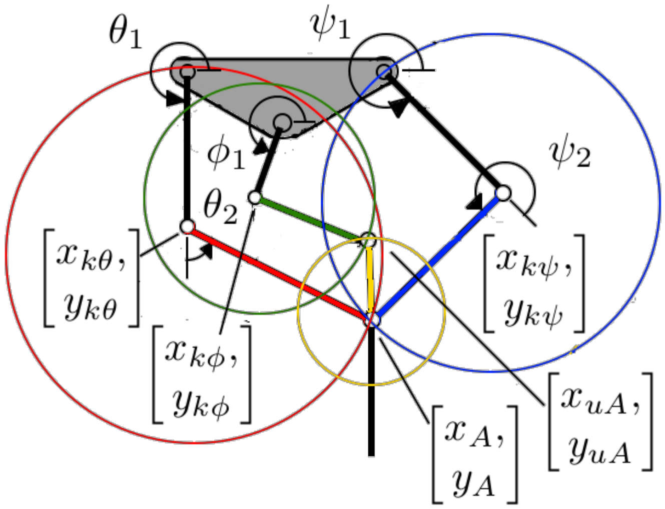
θ_1
ψ_1
ϕ_1
ψ_2
θ_2
$\begin{bmatrix} x_{k\theta}, \\ y_{k\theta} \end{bmatrix}$
$\begin{bmatrix} x_{k\psi}, \\ y_{k\psi} \end{bmatrix}$
$\begin{bmatrix} x_{k\phi}, \\ y_{k\phi} \end{bmatrix}$
$\begin{bmatrix} x_{uA}, \\ y_{uA} \end{bmatrix}$
$\begin{bmatrix} x_A, \\ y_A \end{bmatrix}$

Figure 4.3: The forward kinematics can be solved by considering the intersection of circles.

Using the link lengths and angle conventions defined earlier, the first of these intersection points, (x_A, y_A), can be calculated:

$$\begin{bmatrix} x_A \\ y_A \end{bmatrix} = \begin{bmatrix} \frac{b}{a}\left(x_{k\psi} - x_{k\theta}\right) \pm \frac{c}{a}\left(y_{k\psi} - y_{k\theta}\right) + x_{k\theta} \\ \frac{b}{a}\left(y_{k\psi} - y_{k\theta}\right) \mp \frac{c}{a}\left(x_{k\psi} - x_{k\theta}\right) + x_{k\theta} \end{bmatrix} \text{, where} \tag{4.5}$$

- $(x_{k\theta}, y_{k\theta})$, $(x_{k\phi}, y_{k\phi})$, and $(x_{k\psi}, y_{k\psi})$ are calculated from $q_a = (\theta_1, \phi_1, \psi_1)$,
- $a = \sqrt{\left(x_{k\theta} - x_{k\psi}\right)^2 + \left(y_{k\theta} - y_{k\psi}\right)^2}$,
- $b = \frac{l_2^2 - l_6^2 + a^2}{2a} = \frac{l_2}{l_6}$ (for symmetric links, i.e. when $l_2 = l_6$), and
- $c = \sqrt{l_2^2 - b^2}$.

The same approach can be applied to find the other intersection point, (x_{uA}, y_{uA}). Combining these steps gives Algorithm 1, a compact geometric forward kinematic algorithm:

Observe that equation 4.5 has one ± and one ∓ operator. Currently, these are implemented as `switch` statements, but they introduce jump discontinuities into the forward kinematic equations. Given the limited workspace of the robot, these discontinuities are encountered very rarely, but they warrant further analysis and possibly a more considered approach to the forward kinematics problem.

Algorithm 1 Analytic geometric closed-chain forward kinematics

```
function GEOMFK($q_a$)
    Given $\theta_1$ and $\psi_1$, solve for $(x_A, y_A)$.
    Given $\phi_1$ and $(x_A, y_A)$, solve for $(x_{uA}, y_{uA})$.
    Given $(x_A, y_A)$ and $(x_{uA}, y_{uA})$, solve for $(x_F, y_F, \measuredangle_F)$.
    Use subchain inverse kinematics to solve for $q_u$.
    return $T_{bf} = (x_f, y_f, \measuredangle_f)$ and $q_u$.
end function
```

Numerical forward kinematics

Given $q_a \in \mathbb{R}^3$, the loop-closure equations can be solved directly to obtain $q_u \in \mathbb{R}^6$. Then, once all the joint positions are known, we can choose any of the three open sub-chains and use the corresponding forward kinematics equations to calculate the end-effector pose.

This approach requires solving a system of nonlinear equations (with potentially many solutions), so a numerical root-finding algorithm may be more appropriate than seeking an analytical solution. The simplest such algorithm is the Newton-Raphson method.

Consider a nonlinear equation $g(q) = \dots$, for which we seek the roots, i.e., we wish to solve $g(q) = 0$. Linearize the equation by writing a Taylor series expansion of $g(q)$ about the point q_0:

$$g(q) = g(q_0) + \left.\frac{\partial g}{\partial q}\right|_{q_0} (q - q_0) + \text{h.o.t.} \tag{4.6}$$

Discarding the higher-order terms, the roots are approximately

$$q = q_0 - \left(\left.\frac{\partial g}{\partial q}\right|_{q_0} \right)^{-1} g(q_0) \quad . \tag{4.7}$$

Taking the new q as q_0, this method can be iterated until some convergence threshold criterion is satisfied.

It turns out that $\frac{\partial g}{\partial q}$ is the *constraint Jacobian* (derived in the next section), which shows up often in the analysis of closed kinematic chains. We can use it to develop Algorithm 2, an iterative numerical forward kinematics algorithm.

Algorithm 2 Newton-Raphson closed-chain forward kinematics

```
function NRFK($q_a, q_u^0, \varepsilon$)
    Initialize $r > \varepsilon$                                   ▷ measure of convergence
    while $r > \varepsilon$ do                                      ▷ convergence threshold
        $q_u^{new} = q_u^0 - J_c^{-1} g\left(q_a, q_u^0\right)$
        $r = \frac{\left|g(q_a, q_u^0) - g(q_a, q_u^{new})\right|}{\left|g(q_a, q_u^0)\right|}$      ▷ update measure of convergence
        $q_u^0 = q_u^{new}$
    end while                              ▷ we now have approximate values for $q_u$
    $\theta = \begin{bmatrix} q_a(1) & q_u^{new}(1) & q_u^{new}(2) \end{bmatrix}^{\mathrm{T}}$
    return $\begin{bmatrix} x_f & y_f & \measuredangle_f \end{bmatrix}^{\mathrm{T}} = T_{bf}(\theta)$    ▷ $T_{bf}(\theta)$ is the end-effector pose computed using the $\theta$-chain
end function
```

In summary, this algorithm finds approximate unactuated joint positions from the actuated joint positions and uses the positions along one open sub-chain to compute the end-effector pose. Note that this algorithm uses the θ-chain to compute the end-effector pose, but the other open sub-chains are equally viable.

Pros and Cons of `geomFK` and `NRFK`

Each of the two forward kinematics methods described above has strengths and weaknesses, especially when considering their implementation on an embedded system. Iterative methods do not provide guaranteed execution times, so they are usually poor choices for real-time control. Additionally, matrix inversion is a computationally intensive task with unbounded execution time, so it is best avoided or at least replaced with LU decomposition.

Algorithm	Pros	Cons
`geomFK`	bounded execution time	jump discontinuities
`NRFK`	no jump discontinuities	unbounded execution time & matrix inversion

Based on these pros and cons, and considering that Hopp3r's relatively small workspace generally avoids jump discontinuities, we opted to implement `geomFK` instead of `NRFK`.

4.1.3 Inverse Kinematics

The inverse kinematics problem is to find joint angles that yield a particular end-effector pose (the foot pose, in the case of this robot). Whereas the forward kinematics for this robot (and closed-chain linkages in general) is difficult to compute, the inverse kinematics can be solved analytically, using basic geometry. In fact, for this particular robot, the foot pose $(x_f, y_f, \measuredangle_f)^{\mathrm{T}}$ itself provides a lot of information: from it, the location of the most distal joint (the "ankle") , with respect to the base frame $\{b\}$, can be calculated as

$$\begin{bmatrix} x_A \\ y_A \end{bmatrix} = \begin{bmatrix} x_f - l_8 \cos \measuredangle_f \\ y_f - l_8 \sin \measuredangle_f \end{bmatrix} , \tag{4.8}$$

and similarly, the location of the joint above (the "upper ankle") is

$$\begin{bmatrix} x_{uA} \\ y_{uA} \end{bmatrix} = \begin{bmatrix} x_f - (l_7 + l_8) \cos \measuredangle_f \\ y_f - (l_7 + l_8) \sin \measuredangle_f \end{bmatrix} . \tag{4.9}$$

These two points (the ankle and the upper ankle) are crucial to solving the inverse kinematics because they connect the open subchains, forming closed loops.

θ-chain inverse kinematics

Knowing the ankle's location, we can analytically solve for θ_1 and θ_2 by applying the same geometric inverse kinematics used with 2R robot arms. Figure 4.4 below displays most of the geometry relevant for computing θ_1 and θ_2.

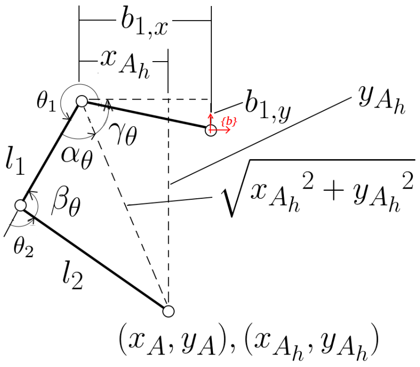

Figure 4.4: Geometric features relevant for computing inverse kinematics of the θ-chain

First, relative to $\{b\}$, the location of joint θ_1 (the "θ hip") is simply

$$\begin{bmatrix} x_{H,\theta} \\ y_{H,\theta} \end{bmatrix} = \begin{bmatrix} -b_1, x \\ b_1, y \end{bmatrix} .$$

To apply 2R inverse kinematics, we also need the location of the angle with respect to the θ hip:

$$\begin{bmatrix} x_{A_{h,\theta}} \\ y_{A_{h,\theta}} \end{bmatrix} = \begin{bmatrix} -x_{H,\theta} + x_A \\ y_{H,\theta} - y_A \end{bmatrix} .$$

If we draw a line connecting the ankle and the θ hip, the angle between this line and the base frame x-axis is simply

$$\gamma_\theta = \operatorname{atan2}(y_{A_{h,\theta}}, x_{A_{h,\theta}}).$$

Now, the angle between this same line and link l_1 can be found using the Law of Cosines:

$$\alpha_\theta = \arccos\left(\frac{x_{A_{h,\theta}}^2 + y_{A_{h,\theta}}^2 + l_1^2 - l_2^2}{2l_1\sqrt{x_{A_{h,\theta}}^2 + y_{A_{h,\theta}}^2}}\right),$$

and the hip angle is

$$\theta_1 = -\gamma_\theta - \alpha_\theta\ . \tag{4.10}$$

The (x, y) location of the θ_2 joint (the "knee") can now be computed:

$$\begin{bmatrix} x_{K,\theta} \\ y_{K,\theta} \end{bmatrix} = \begin{bmatrix} x_{H,\theta} + l_1 \cos\theta_1 \\ y_{H,\theta} + l_1 \sin\theta_1 \end{bmatrix}\ .$$

We can also find θ_2 by applying the Law of Cosines again:

$$\beta_\theta = \arccos\left(\frac{l_1^2 + l_2^2 - x_{A_{h,\theta}}^2 - y_{A_{h,\theta}}^2}{2l_1l_2}\right)\ ,$$

and the knee angle is simply

$$\theta_2 = \pi - \beta_\theta\ . \tag{4.11}$$

We can verify the inverse kinematics result by calculating the (x, y) location of the ankle using θ_1 and θ_2:

$$\begin{bmatrix} x_{A,\theta} \\ y_{A,\theta} \end{bmatrix} = \begin{bmatrix} x_{H,\theta} + l_1 \cos\theta_1 + l_2 \cos\theta_1 + \theta_2 \\ y_{H,\theta} + l_1 \sin\theta_1 + l_2 \sin\theta_1 + \theta_2 \end{bmatrix}\ .$$

Lastly, because we know θ_1, θ_2, and $\measuredangle_f$, we can find θ_3:

$$\theta_3 = \measuredangle_f - \theta_1 - \theta_2\ . \tag{4.12}$$

ψ-chain inverse kinematics

The inverse kinematics of the ψ-chain mirror those of the θ-chain. Figure 4.5 below shows the relevant geometric features of the ψ-chain.

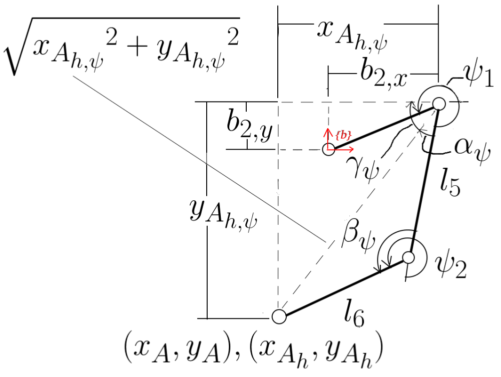

Figure 4.5: Geometric features relevant for computing inverse kinematics of the ψ-chain

In the $\{b\}$ frame, the origin of the ψ-chain is

$$\begin{bmatrix} x_{H,\psi} \\ y_{H,\psi} \end{bmatrix} = \begin{bmatrix} b_2, x \\ b_2, y \end{bmatrix}\ ,$$

and with respect to this point, the location of the "ankle" joint is

$$\begin{bmatrix} x_{A_{h,\psi}} \\ y_{A_{h,\psi}} \end{bmatrix} = \begin{bmatrix} x_{H,\psi} - x_A \\ y_{H,\psi} - y_A \end{bmatrix} .$$

The "hip" angle, ψ_1, is given by

$$\gamma_\psi = \text{atan2}(y_{A_{h,\psi}}, x_{A_{h,\psi}}),$$

$$\alpha_\psi = \arccos\left(\frac{x^2_{A_{h,\psi}} + y^2_{A_{h,\psi}} + l_5^2 - l_6^2}{2l_5\sqrt{x^2_{A_{h,\psi}} + y^2_{A_{h,\psi}}}}\right) ,$$

$$\psi_1 = \pi + \gamma_\psi + \alpha_\psi \ . \tag{4.13}$$

The "knee" angle, ψ_2, is given by

$$\beta_\psi = \arccos\left(\frac{l_5^2 + l_6^2 - x^2_{A_{h,\psi}} - y^2_{A_{h,\psi}}}{2l_5 l_6}\right) ,$$

$$\psi_2 = \pi + \beta_\psi \ . \tag{4.14}$$

The "ankle" angle, ψ_3, is simply

$$\psi_3 = \measuredangle_f - \psi_1 - \psi_2 \ . \tag{4.15}$$

ϕ-chain inverse kinematics

Just as a 2R arm has "elbow-up" and "elbow-down" configurations, this planar robot has a number of possible configurations for the same end-effector pose. One such example is the "elbow-left" or "elbow-right" configuration of the ϕ chain. The derivations below assume the "elbow-left" configuration, so the equations resemble those of the θ chain (the "elbow-right" configuration equations resemble those of the ψ-chain).

This model defines the origin of the $\{b\}$ frame at joint ϕ_1, so

$$\begin{bmatrix} x_{H,\phi} \\ y_{H,\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} .$$

The "ankle" of the ϕ chain is different from that of the θ and ψ chains and is consequently referred to as the "upper ankle". Repeating the same geometric procedure as done above for the other two chains, we first calculate the "hip" angle, ϕ_1:

$$\gamma_\phi = \text{atan2}(y_{uA}, x_{uA}),$$

$$\alpha_\phi = \arccos\left(\frac{x^2_{uA} + y^2_{uA} + l_3^2 - l_4^2}{2l_3\sqrt{x^2_{uA} + y^2_{uA}}}\right) ,$$

$$\phi_1 = -\gamma_\phi - \alpha_\phi \ . \tag{4.16}$$

The "knee" angle, ϕ_2, is given by

$$\beta_\phi = \arccos\left(\frac{l_3^2 + l_4^2 - x^2_{uA} - y^2_{uA}}{2l_3 l_4}\right) ,$$

$$\phi_2 = \pi - \beta_\phi \ . \tag{4.17}$$

The "ankle" angle, ϕ_3, is simply

$$\phi_3 = \measuredangle_f - \phi_1 - \phi_2 \ . \tag{4.18}$$

4.1.4 Actuator Jacobian

We would now like to solve the inverse dynamics problem: what motor speeds/torques are required to generate a desired end-effector speed/force? The solution can be approximated using the *actuator Jacobian.*

Loop-closure equations

Refer to Figures 4.1 and 4.2 and observe that the open sub-chains intersect at a common joint, measured by θ_3 and ψ_3. Consequently, there are three loops: the θ-ϕ loop, the ϕ-ψ loop, and the ψ-θ loop. We can describe these three loops with two sets of equations:

$$g(\theta, \phi, \psi) := \begin{bmatrix} T_{bf}(\theta) - T_{bf}(\phi) \\ -T_{bf}(\phi) + T_{bf}(\psi) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.19}$$

where $T_{bf}(\cdot)$ represents the twist from the base frame $\{b\}$ to the end-effector frame $\{f\}$ along the corresponding open sub-chain; this is just a compact representation of the earlier open sub-chain forward kinematic equations, so the left and right sides of the equation are 6×1 column vectors.

For a given set of actuated joint angles $q_a = [\theta_1, \phi_1, \psi_1]^{\mathrm{T}}$, we seek unactuated joint angles ($q_u = [\theta_2, \theta_3, \phi_2, \phi_3, \psi_2, \psi_3]^{\mathrm{T}}$), i.e., the roots of the loop-closure equations, so the iterative Newton-Raphson root-finding equation is

$$\left[\theta_2^{k+1}, \theta_3^{k+1}, \phi_2^{k+1}, \phi_3^{k+1}, \psi_2^{k+1}, \psi_3^{k+1}\right]^{\mathrm{T}} = q_u^{k+1} = q_u^k - J_c^{-1} g\left(q_a, q_u^k\right) \ , \tag{4.20}$$

where J_c, the *constraint Jacobian*, is the matrix of partial derivatives of the loop-closure expression with respect to the elements of q_u. Since the loop-closure equation was expressed as a 6×1 column vector, the constraint Jacobian is a square 6×6 matrix.

Constraint Jacobian

As mentioned earlier, the constraint Jacobian J_c is the matrix of partial derivatives of the loop-closure expression with respect to the unactuated joint positions q_u. Differentiating the loop-closure equation above, we get the following 6×6 matrix:

$$J_c = \begin{bmatrix} \frac{\partial T_{bf}(\theta)}{\partial \theta_2} & \frac{\partial T_{bf}(\theta)}{\partial \theta_3} & -\frac{\partial T_{bf}(\phi)}{\partial \phi_2} & -\frac{\partial T_{bf}(\phi)}{\partial \phi_3} & 0 & 0 \\ 0 & 0 & -\frac{\partial T_{bf}(\phi)}{\partial \phi_2} & -\frac{\partial T_{bf}(\phi)}{\partial \phi_3} & \frac{\partial T_{bf}(\psi)}{\partial \psi_2} & \frac{\partial T_{bf}(\psi)}{\partial \psi_3} \end{bmatrix} . \tag{4.21}$$

Actuator Jacobian derivation

The actuator Jacobian maps generalized actuator velocities and forces to end-effector velocities and forces. Returning to the loop-closure equation (equation 4.19) and differentiating with respect to all joint positions (q_a and q_u), we get

$$\begin{bmatrix} J_\theta & -J_\phi & 0 \\ 0 & -J_\phi & J_\psi \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = 0_{6\times 1} \ , \tag{4.22}$$

where each $J \in \mathbb{R}^{3\times 3}$ is the Jacobian of the corresponding open sub-chain. Expanding this equation is straightforward and will help with the next step. When fully expanded, equation 4.22 becomes

$$\begin{bmatrix} J_{\theta_{1,1}} & J_{\theta_{1,2}} & J_{\theta_{1,3}} & -J_{\phi_{1,1}} & -J_{\phi_{1,2}} & -J_{\phi_{1,3}} & 0 & 0 & 0 \\ J_{\theta_{2,1}} & J_{\theta_{2,2}} & J_{\theta_{2,3}} & -J_{\phi_{2,1}} & -J_{\phi_{2,2}} & -J_{\phi_{2,3}} & 0 & 0 & 0 \\ J_{\theta_{3,1}} & J_{\theta_{3,2}} & J_{\theta_{3,3}} & -J_{\phi_{3,1}} & -J_{\phi_{3,2}} & -J_{\phi_{3,3}} & 0 & 0 & 0 \\ 0 & 0 & 0 & -J_{\phi_{1,1}} & -J_{\phi_{1,2}} & -J_{\phi_{1,3}} & J_{\psi_{1,1}} & J_{\psi_{1,2}} & J_{\psi_{1,3}} \\ 0 & 0 & 0 & -J_{\phi_{2,1}} & -J_{\phi_{2,2}} & -J_{\phi_{2,3}} & J_{\psi_{2,1}} & J_{\psi_{2,2}} & J_{\psi_{2,3}} \\ 0 & 0 & 0 & -J_{\phi_{3,1}} & -J_{\phi_{3,2}} & -J_{\phi_{3,3}} & J_{\psi_{3,1}} & J_{\psi_{3,2}} & J_{\psi_{3,3}} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \\ \dot{\psi}_1 \\ \dot{\psi}_2 \\ \dot{\psi}_3 \end{bmatrix} = 0_{6\times 1} \ , \tag{4.23}$$

which can be rearranged into an actuated part, $H_a \in \mathbb{R}^{6\times 3}$, and an unactuated part, $H_u \in \mathbb{R}^{6\times 6}$, such that

$$\begin{bmatrix} H_a(q_a, q_u) & H_u(q_a, q_u) \end{bmatrix} \begin{bmatrix} \dot{q}_a \\ \dot{q}_u \end{bmatrix} = 0_{6\times 1} \ , \tag{4.24}$$

or equivalently,

$$\dot{q}_u = -H_u^{-1} H_a \dot{q}_a \ . \tag{4.25}$$

From equation 4.23, it is apparent that

$$H_a(q_a, q_u) = \begin{bmatrix} J_{\theta_{1,1}} & -J_{\phi_{1,1}} & 0 \\ J_{\theta_{2,1}} & -J_{\phi_{2,1}} & 0 \\ J_{\theta_{3,1}} & -J_{\phi_{3,1}} & 0 \\ 0 & -J_{\phi_{1,1}} & J_{\psi_{1,1}} \\ 0 & -J_{\phi_{2,1}} & J_{\psi_{2,1}} \\ 0 & -J_{\phi_{3,1}} & J_{\psi_{3,1}} \end{bmatrix} \text{, and} \tag{4.26}$$

$$H_u(q_a, q_u) = \begin{bmatrix} J_{\theta_{1,2}} & -J_{\phi_{1,3}} & -J_{\phi_{1,2}} & -J_{\phi_{1,3}} & 0 & 0 \\ J_{\theta_{2,2}} & -J_{\phi_{2,3}} & -J_{\phi_{2,2}} & -J_{\phi_{2,3}} & 0 & 0 \\ J_{\theta_{3,2}} & -J_{\phi_{3,3}} & -J_{\phi_{3,2}} & -J_{\phi_{3,3}} & 0 & 0 \\ 0 & 0 & -J_{\phi_{1,2}} & -J_{\phi_{1,3}} & J_{\psi_{1,2}} & J_{\psi_{1,3}} \\ 0 & 0 & -J_{\phi_{2,2}} & -J_{\phi_{2,3}} & J_{\psi_{2,2}} & J_{\psi_{2,3}} \\ 0 & 0 & -J_{\phi_{3,2}} & -J_{\phi_{3,3}} & J_{\psi_{3,2}} & J_{\psi_{3,3}} \end{bmatrix} , \tag{4.27}$$

which is, in fact, the constraint Jacobian J_c from the previous section (see equation 4.21), so the velocities of the unactuated joints are

$$\dot{q}_u = -J_c^{-1} H_a \dot{q}_a \ , \tag{4.28}$$

assuming J_c is invertible. Observe that $-J_c^{-1}H_a$ is a 6×3 matrix that maps actuated joint velocities to unactuated joint velocities. For example,

$$\dot{\theta}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \dot{q}_a \text{ , and} \tag{4.29}$$

$$\dot{\theta}_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \dot{q}_a \ . \tag{4.30}$$

We can develop a mapping from the velocities of the actuated joints to the end-effector velocity, using the θ-chain:

$$\begin{bmatrix} \dot{x}_f \\ \dot{y}_f \\ \dot{\measuredangle}_f \end{bmatrix} = J_\theta\left(q_a, q_u\right)\dot{\theta} = J_\theta \begin{bmatrix} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \end{bmatrix} \dot{q}_a \ , \tag{4.31}$$

or more simply,

$$\begin{bmatrix} \dot{x}_f \\ \dot{y}_f \\ \dot{\measuredangle}_f \end{bmatrix} = J_{a,\theta}\left(q_a, q_u\right)\dot{q}_a \ , \tag{4.32}$$

where $J_{a,\theta} \in \mathbb{R}^{3\times 3}$ is the θ-chain *actuator Jacobian*. Similar expressions can be derived for $J_{a,\phi} \in \mathbb{R}^{3\times 3}$ and $J_{a,\psi} \in \mathbb{R}^{3\times 3}$ by returning to equation 4.28 and deriving expressions for $\dot{\phi}_2$, $\dot{\phi}_3$, $\dot{\psi}_2$, and $\dot{\psi}_3$. Together, the actuator Jacobians are

$$J_{a,\theta} = J_\theta \begin{bmatrix} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \end{bmatrix} \tag{4.33}$$

$$J_{a,\phi} = J_\phi \begin{bmatrix} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \end{bmatrix} \tag{4.34}$$

$$J_{a,\psi} = J_{\psi} \begin{bmatrix} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \left(-J_c^{-1} H_a\right) \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \left(-J_c^{-1} H_a\right) \end{bmatrix} . \tag{4.35}$$

The relationship between joint speeds and end-effector speed is given above in equation 4.32, but in general, the relationship is

$$V = J_a \dot{\theta} \quad \text{and } \dot{\theta} = J_a^{-1} V \ . \tag{4.36}$$

Similarly, the relationship between joint torques and end-effector force is

$$\tau = J_a^{\mathrm{T}} F \quad \text{and } F = J_a^{\text{-}\mathrm{T}} \tau \tag{4.37}$$

4.1.5 Link geometry optimization for jumping

Having developed the forward and inverse kinematics and the actuator Jacobian, it is now possible to optimize link geometry for virtually compliant locomotion. Consider two limit cases:

1. if the robot is too small, the distance over which a virtual spring can compress is too short, so the robot is not very compliant;

2. conversely, if the robot is too large, generating an arbitrary end-effector wrench will require enormous motor torques.

One way to optimize link geometry for virtually compliant locomotion is to maximize both the linkage's mechanical advantage and the end-effector's workspace. The workspace is the reachable subset of the task space ($\mathcal{W} \in SE(2)$). Maximizing the linkage's mechanical advantage allows selection of a lower-torque motor to apply a given end-effector wrench. Maximizing the "volume" of the end-effector's workspace is important not just because it maximizes not only the robot's mobility but also the "travel" over which a virtual spring can compress and extend.

The nonlinear kinematic equations and actuator Jacobian result in a feature-rich parameter space with potentially many local minima, so a nonlinear optimization method such as simulated annealing or MATLAB's `fmincon` is more suitable than gradient descent for finding the optimizer. After selecting an optimization method, the problem reduces to designing an objective function that captures the qualities of the optimizer (large mechanical advantage and large workspace). Algorithm 3 computes the value of such a quadratic objective function that captures these two qualities.

Algorithm 3 Link length objective function

```
function ObjectiveFunction($l$)                 ▷ $l$ is a list of the leg lengths and actuated joint locations.
    for each $(x_{f,i}, y_{f,i}, \measuredangle_{f,i}) \in \mathcal{W}$ do
        if IK solvable & $q_{min} \leq q \leq q_{max}$ then
            append $(x_{f,i}, y_{f,i}, \measuredangle_{f,i})$ to $\mathcal{W}_{reachable}$
            Given $l, (x_{f,i}, y_{f,i}, \measuredangle_{f,i}), q$, compute $\tau = J_a^{\mathrm{T}} F$
            append $\|\tau\|$ to $\tau_{list}$
        end if
        $V_{workspace} = \mathrm{vol}(\mathcal{W}_{reachable})$
        $\tau_{max} = \max(\tau_{list})$
    end for
    return $J = w_1 \tau_{max}^2 + w_2 V_{workspace}^2$.
end function
```

4.2 Motor selection

4.2.1 Torque and speed requirements

4.2.2 Thermal analysis

Thermal model

$$\frac{d}{dt}\begin{bmatrix} T_w \\ T_h \\ T_{AL} \end{bmatrix} = \begin{bmatrix} -\frac{1}{m_w c_w R_{T,wh}} & \frac{1}{m_w c_w R_{T,wh}} & 0 \\ \frac{1}{m_h c_h R_{T,wh}} & -\frac{1}{m_h c_h R_{T,wh}} - \frac{1}{m_h c_h R_{T,hAL}} & \frac{1}{m_h c_h R_{T,hAL}} \\ 0 & \frac{1}{m_{AL} c_{AL} R_{T,hAL}} & -\frac{1}{m_{AL} c_{AL} R_{T,hAL}} - \frac{h_{AL} A_{AL}}{m_{ALh} c_{AL}} \end{bmatrix} \begin{bmatrix} T_w \\ T_h \\ T_{AL} \end{bmatrix} + \begin{bmatrix} \frac{I^2 R_w}{m_w c_w} \\ 0 \\ \frac{h_{AL} A_{AL}}{m_{AL} c_{AL}} T_{amb} \end{bmatrix} \tag{4.38}$$

Simulation results

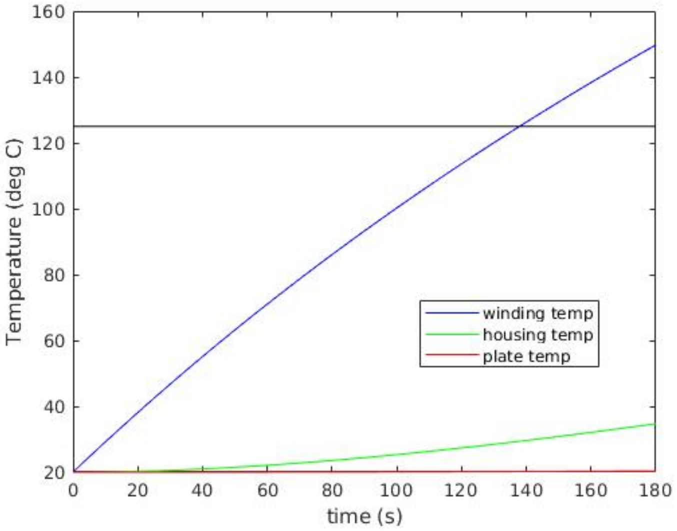

Figure 4.6: Temperature profiles of motor winding, motor housing, and backplate during a balancing task

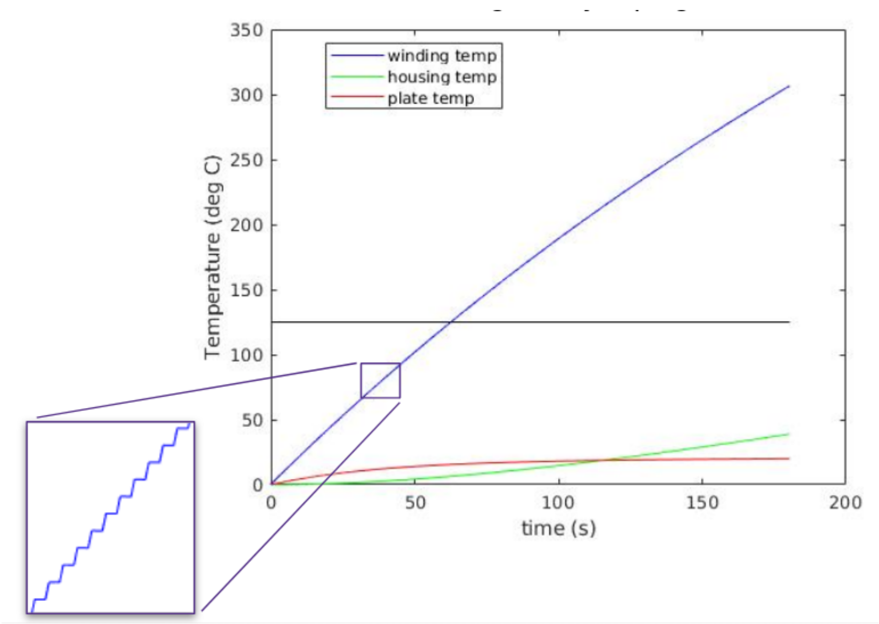

Figure 4.7: Temperature profiles of motor winding, motor housing, and backplate during steady state jumping. The close-up shows the sharp rise in temperature during each jump.

Observability

4.3 Shaft analysis

4.4 Boom selection

4.5 Counterweight selection

4.6 Acrobot framework for a balancing controller

4.6.1 Lagrangian dynamics

4.6.2 Linearized model

4.6.3 LQR balancing controller

4.6.4 Translation to Hopp3r

Chapter 5

Design Evolution

5.1 Mechanical

5.2 Electrical

5.3 Software

Chapter 6

Final Design Description

6.1 Mechanical

6.2 Electrical

6.3 Software

Chapter 7

Testing Procedures

Chapter 8

Performance Results

Chapter 9

Suggested Next Steps

9.1 Mechanical

9.2 Electrical

9.3 Software

Chapter 10

Other Documentation

Bibliography

[1] K. M. Lynch and F. C. Park. *Modern Robotics.* Cambridge University Press.