Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE 2018
August 26-29, 2018, Quebec City, Quebec, Canada

DETC2018-86048

DYNAMIC MODELING OF A QUADRUPED WITH A ROBOTIC TAIL USING VIRTUAL WORK PRINCIPLE

Yujiong Liu
Mechanical Engineering Department
Robotics and Mechatronics Laboratory
Virginia Tech
Blacksburg, VA, USA

Pinhas Ben-Tzvi
Mechanical Engineering Department
Robotics and Mechatronics Laboratory
Virginia Tech
Blacksburg, VA, USA
bentzvi@vt.edu

ABSTRACT

For utilizing robotic tail to stabilize and maneuver a quadruped, it is important to understand the mechanism of how the tail motion influences the quadruped motion which requires obtaining an analytic dynamic model. This paper presents a systematic methodology for modeling the dynamics of a general quadruped (capable of all 6 DOF motions) with a robotic pendulum tail based on the virtual work principle. The formulation of this model is motivated by robotic tail research, it can also be used as an alternative approach to model the quadruped dynamics other than using Lagrangian and Newton-Euler based methods. Numerical simulations are also conducted to verify both the forward and the inverse model.

1 INTRODUCTION

Recently, inspired by animals, researchers became interested in using robotic tails [1-8] to help maneuver and stabilize the locomotion [9] of bipedal and quadrupedal robots. In order to achieve this, understanding the mechanism of how the tail motion influences the body motion, i.e., the dynamic model of legged robots with robotic tails, is necessary. For the tail-body dynamics, most researchers either use simple models ([1, 4] consider only the plenary motion for modeling while [2, 4, 6] treating the robot as one rigid body) or model the dynamics based on the assumption that the feet are able to slide on the ground [3]. This simplification is effective for a specific robot [3, 4] or for the case that the robot is in the air [2]. However, when the robot is walking on the ground, the effect of the foot-ground friction cannot be ignored. In this case, forcing the tail to act may cause the robot to fall over. On the other hand, the simplified planar model is not always valid, especially during fast motion. The effect of the leg motions may have a significant contribution to the whole body dynamics. All these motivated us to develop a new dynamic model that allows all 6DOF motions of the quadruped and does not rely on the sliding assumption.

At present, most dynamical models of the quadruped are based on the Lagrangian equation or the Newton-Euler equations. Newton-Euler method [10] computes the dynamic terms recursively, which is very suitable to implement numerically. However, this method requires computing unnecessary constraint forces and is hard to investigate the dynamical system analytically. Lagrangian method [11, 12] is another popular way to formulate the dynamic model. However, the Lagrangian method requires computing the derivative of kinetic energy, which is normally hard to process, especially for systems with high dimensions and closed kinematic chains. In addition, to make the formulation easier, more generalized coordinates (the so-called cyclic coordinates) than the system degrees of freedom are used. Thus, to make the system determinate, additional constraint equations along with the differential equation are required, which yields a hard-to-solve differential algebraic equation (DAE).

Other approaches proposed in the literature include Center of Inertia (COI) [13], floating base method [14], modular formulation frame [15] and virtual power [16] (Kane's method). However, the basic methodologies these approaches adopt also belong to the above two approaches.

Virtual work principle is another popular approach to derive the dynamic model of a multibody system. This method has the benefit of eliminating the constraint forces, which makes it very useful when researchers are only concerned about the overall motions and the actuation forces. In general, quadruped is such a case. Moreover, a quadruped can be regarded as a parallel mechanism. It is well known that for a parallel mechanism, the inverse dynamics are relatively easy while the forward dynamics are harder to derive. Thus, virtual work principle is widely used in parallel mechanisms [17, 18] to

formulate the inverse dynamics. On the other hand, due to the nature of virtual work (assembling the whole body dynamics by adding up the part dynamics), it is potentially useful to model the dynamics modularly. In addition, it is well known that the hybrid dynamics of legged robots usually have several, sometimes even more than ten phases. A unified way to model the different phases would be very useful for the research community. All these motivate us to use virtual work principle to model the quadruped dynamics. To the authors' best knowledge, there is currently no dynamics modelling done by virtual work for quadruped robots.

Note that this paper focuses on the usage of virtual work principle in the dynamic modeling of the quadruped. Therefore, we only considered the case that all four legs of the quadruped are on the ground. The rest of this paper is organized as follows. Section 2 presents the kinematic analysis of the mechanism, which includes position analysis, velocity analysis, and acceleration analysis. Section 3 introduces the virtual work principle briefly and formulates the inverse and forward dynamics, respectively. Finally, two numerical simulations are presented in section 4 to validate both the inverse and forward models.

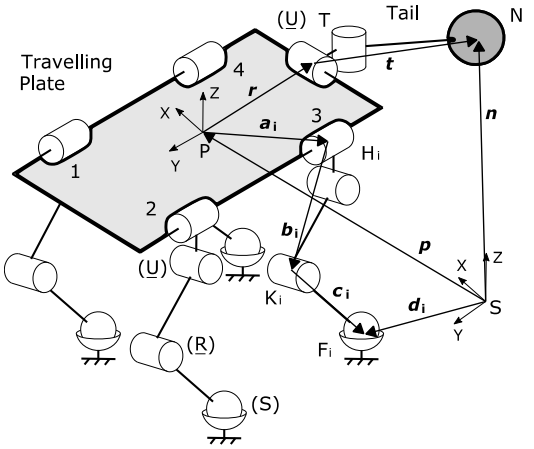

Figure 1. The kinematic configuration of a quadruped with a tail

2 KINEMATIC ANALYSIS

The kinematic configuration of the quadruped is shown in Fig. 1. The quadruped consists of one torso, one tail, and four identical legs. In this paper, since we only consider the dynamics when the four feet are all touching the ground, the quadruped essentially can be regarded as a hybrid mechanism such that the torso with four legs constitutes the parallel mechanism part, and the torso with the tail constitutes the serial mechanism part. Therefore, the torso is called the traveling plate for the traditional parallel mechanism. Each leg consists of a $\underline{SRRR}$ kinematic chain in which the universal joint on the hip is decomposed into two intersecting revolute joints and the feet are modeled as spherical joints.

The tail is a massless bar with a point mass on the tip and is actuated by a universal joint connected to the body. Since there are 8 DOF's for this quadruped (6 DOF's for the parallel mechanism and 2 DOF's for the tail), we choose $\mathbf{q} = [\mathbf{p}^T \ \phi_x \ \phi_y \ \phi_z \ \theta_{ta} \ \theta_{tb}]^T$ as the independent generalized coordinate set where $\mathbf{p}$ is the position vector of the travelling plate center, $\phi_x \ \phi_y \ \phi_z$ are the rotational angles of the travelling plate with respect to the x y and z axis of the global frame, respectively. θ_{ta} and θ_{tb} are the rotational angle of the tail with respect to the travelling plate.

To define these variables accurately, inertial frame ΣS is attached on the ground. Body fixed frame ΣP of the travelling plate is attached on the travelling plate center P with its initial orientation being the same as frame ΣS. $\mathbf{R}_P^S$ is the rotation matrix from frame ΣP to ΣS. The rotation matrix $\mathbf{R}_P^S$ is defined by the roll, pitch, and yaw angles that is, rotating ϕ_x about the fixed x-axis first, then rotating ϕ_y about the fixed y-axis, finally rotating ϕ_z about the fixed z-axis. Thus, the rotation matrix is

$$\mathbf{R}_P^S = \begin{bmatrix} c\phi_z c\phi_y & c\phi_z s\phi_y s\phi_x - s\phi_z c\phi_x & c\phi_z s\phi_y c\phi_x + s\phi_z s\phi_x \\ s\phi_z c\phi_y & s\phi_z s\phi_y s\phi_x + c\phi_z c\phi_x & s\phi_z s\phi_y c\phi_x - c\phi_z s\phi_x \\ -s\phi_y & c\phi_y s\phi_x & c\phi_y c\phi_x \end{bmatrix} \quad (1)$$

Then the angular velocity and angular acceleration of the quadruped body are

$$\boldsymbol{\omega} = \begin{bmatrix} \dot{\phi}_x & \dot{\phi}_y & \dot{\phi}_z \end{bmatrix}^T \quad (2)$$

$$\dot{\boldsymbol{\omega}} = \begin{bmatrix} \ddot{\phi}_x & \ddot{\phi}_y & \ddot{\phi}_z \end{bmatrix}^T \quad (3)$$

Since the quadruped is a parallel mechanism, in essence, we can follow a similar procedure utilized in parallel mechanisms to find the necessary kinematic terms for the quadruped.

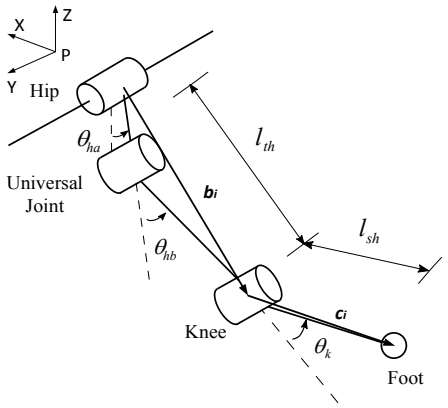

Figure 2. Kinematic parameters of leg *i*

2.1 Position Analysis

This section solves the inverse kinematics of the quadruped. For each leg, the joint angles are defined as in Fig. 2 where $\theta_{ha,i}, \theta_{hb,i} \in (-\pi/2, \pi/2)$ and $\theta_{k,i} \in (0, \pi/2)$. Based on the kinematics configuration and vector definitions in Fig. 1, the vector loop constraint of leg i can be written as

$$\mathbf{b}_i + \mathbf{c}_i = \mathbf{d}_i - \mathbf{p} - \mathbf{a}_i \quad (4)$$

Expressing Eq. (4) in frame ΣP and denoting $\mathbf{s}_i = \mathbf{d}_i^{(P)} - \mathbf{p}^{(P)} - \mathbf{a}_i^{(P)}$ in which $\mathbf{a}_i^{(P)}$ is a constant vector and $\mathbf{d}_i^{(P)} = \mathbf{R}_S^P \mathbf{d}_i$, $\mathbf{p}^{(P)} = \mathbf{R}_S^P \mathbf{p}$, Eq. (4) can be written as

$$\mathbf{b}_i^{(P)} + \mathbf{c}_i^{(P)} = \mathbf{s}_i \tag{5}$$

In this equation, $\mathbf{s}_i$ is a known vector which can be calculated in advance and $\mathbf{b}_i^{(P)}$, $\mathbf{c}_i^{(P)}$ are two vectors containing the three unknown revolute joint angles $\theta_{ha,i}$, $\theta_{hb,i}$ and $\theta_{k,i}$. Thus, Eq. (5) has three equations and three unknowns. Solving this equation yields

$$cos(\pi - \theta_{k,i}) = \frac{\mathbf{s}_i{}^{\mathrm{T}}\mathbf{s}_i - {l_{th}}^2 - {l_{sh}}^2}{2 l_{th} l_{sh}} \tag{6}$$

The rest of the two unknowns can be found successively

$$\theta_{ha,i} = \mathrm{atan2}(s_{ix}, s_{iz}) \tag{7}$$

$$\theta_{hb,i} = \mathrm{atan}(\mathbf{s}'_{iy}/\mathbf{s}'_{iz}) - \mathrm{asin}\left(\frac{2A}{||\mathbf{s}_i|| l_{th}}\right) \tag{8}$$

where $\mathbf{s}'_{iy}$ and $\mathbf{s}'_{iy}$ are the y and z components of $\mathbf{s}'_i = \mathbf{R}_y(\theta_{ha,i})^T \mathbf{s}_i$. $\mathbf{R}_y(\theta_{ha,i})$ denotes the rotation matrix with angle $\theta_{ha,i}$ with respect to the y axis. A is the area of triangle $\Delta H_i K_i F_i$, which can be computed by Heron's formula

$$A = \sqrt{s(s - l_{th,i})(s - l_{sh,i})(s - ||\mathbf{s}_i||)} \tag{9}$$

where

$$s = (l_{th,i} + l_{sh,i} + ||\mathbf{s}_i||)/2$$

The position of the tail can be obtained straightforwardly by

$$\mathbf{n} = \mathbf{p} + \mathbf{r} + \mathbf{t} \tag{10}$$

where

$$\mathbf{r} + \mathbf{t} = \mathbf{R}_P^S(\mathbf{r}^{(P)} + \mathbf{R}_x(\theta_{ta})\mathbf{R}_z(\theta_{tb})[0, -l_t, 0]^T)$$

with parameters defined in Fig. 3. $\mathbf{R}_x(\theta_{ta})$ and $\mathbf{R}_z(\theta_{tb})$ are the principle rotation matrices with respect to x axis and z axis, respectively.

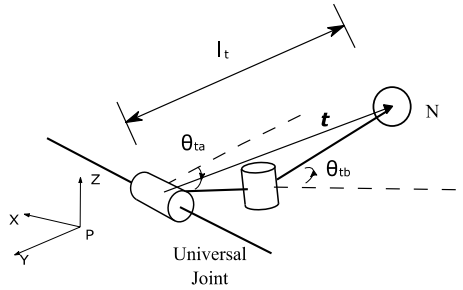

Figure 3. Kinematic parameters of the tail

2.2 Joint Jacobian Matrices

This subsection derives the Jacobian matrices for each actuation joint. These matrices will be needed in subsequent sections. The method used here is direct differentiation of joint angles. Therefore, differentiating Eq. (6) directly yields

$$cos(\pi - \theta_{k,i})\,\dot{\theta}_{k,i} = \frac{\mathbf{s}_i^T \dot{\mathbf{s}}_i}{l_{th} l_{sh}} \tag{11}$$

which requires the differentiation of $\mathbf{s}_i$ first. Since $\mathbf{s}_i$ is expressed in frame ΣP, we need to transform it back to the inertia frame ΣS first. Differentiating $\mathbf{s}_i^{(S)}$ yields

$$\mathbf{R}_P^S \dot{\mathbf{s}}_\iota = -\mathbf{v} + (\mathbf{d}_i - \mathbf{p}) \times \boldsymbol{\omega} \tag{12}$$

where $\mathbf{v}$ is the velocity of point P. Therefore

$$\dot{\mathbf{s}}_\iota = \mathbf{j}_{s,i}\mathbf{t}_p \tag{13}$$

where $\mathbf{j}_{s,i} = [-\mathbf{R}_S^P \quad \mathbf{R}_S^P(\mathbf{d}_i - \mathbf{p})]$ is the Jacobian matrix for $\mathbf{s}_i$ and $\mathbf{t}_p = [\mathbf{v}^T \quad \boldsymbol{\omega}^T]^T$ is the twist of the travelling plate. Therefore, rearranging Eq. (11) yields

$$\dot{\theta}_{k,i} = \mathbf{j}_{k,i}\mathbf{t}_p \tag{14}$$

where

$$\mathbf{j}_{k,i} = \frac{\mathbf{s}_i{}^T \mathbf{j}_{s,i}}{l_{th} l_{sh} sin\theta_{k,i}} \tag{15}$$

is the Jacobian matrix of the knee joint for leg i. Similarly, Eq. (7) is differentiated to obtain the Jacobian matrix of $\theta_{ha,i}$

$$\mathbf{j}_{ha,i} = \frac{cos^2\theta_{ha,i}}{s_{iz}^2}[s_{iz}, 0, -s_{ix}]\mathbf{j}_{s,i} \tag{16}$$

in which s_{ix} and s_{iz} are the x and z components of $\mathbf{s}_i$, respectively. $\dot{\theta}_{hb,i}$ is more challenging to obtain. Instead of differentiating Eq. (8) directly, the y part of Eq. (5) is a better choice. This yields

$$\dot{\theta}_{hb,i} = \frac{-\dot{s}_{iy} - l_{sh}\,cos(\theta_{hb,i} + \theta_{k,i})\dot{\theta}_{k,i}}{l_{th} cos\theta_{hb,i} + l_{sh}\,cos(\theta_{hb,i} + \theta_{k,i})} \tag{17}$$

Since $\dot{s}_{iy}$ and $\dot{\theta}_{k,i}$ have been obtained in Eq. (13) and Eq. (14), substituting these two terms into Eq. (18) gives the Jacobian matrix for θ_{hb}

$$j_{hb,i} = \frac{-l_{th} sin\theta_{k,i}[0,1,0]\mathbf{j}_{s,i} - cos(\theta_{hb,i} + \theta_{k,i})\,\mathbf{s}_i{}^T \mathbf{j}_{s,i}}{l_{th}^2 sin\theta_{k,i} cos\theta_{hb,i} + l_{sh} l_{th} sin\theta_{k,i}\,cos(\theta_{hb,i} + \theta_{k,i})} \tag{18}$$

2.3 Velocity Analysis and Point Jacobian Matrices

Velocity can be calculated by differentiating the position vector. Therefore, the velocity of point H_i is given by

$$\mathbf{v}_{h,i} = \mathrm{d}(\mathbf{p} - \mathbf{a}_i)/dt = \mathbf{v} + \boldsymbol{\omega} \times \mathbf{a}_i \tag{19}$$

The corresponding Jacobian matrix of point H_i is

$$\mathbf{J}_{h,i} = [I \quad -\tilde{\mathbf{a}}_i \quad \mathbf{0}_{3\times 2}] \tag{20}$$

Velocity of point K_i can be obtained similarly

$$\mathbf{v}_{k,i} = \mathbf{v} - (\mathbf{a}_i + \mathbf{b}_i) \times \boldsymbol{\omega} + \mathbf{R}_P^S \mathbf{Q}_{k,i}[\dot{\theta}_{ha,i} \quad \dot{\theta}_{hb,i}]^T \tag{21}$$

where

$$\mathbf{Q}_{k,i} = l_{th}\begin{bmatrix} -cos\theta_{ha,i}cos\theta_{hb,i} & sin\theta_{ha,i}sin\theta_{hb,i} \\ 0 & -cos\theta_{hb,i} \\ sin\theta_{ha,i}cos\theta_{hb,i} & cos\theta_{ha,i}sin\theta_{hb,i} \end{bmatrix} \quad (22)$$

Thus the Jacobian matrix of point K_i is

$$\mathbf{J}_{k,i} = \begin{bmatrix} [\mathbf{I} & -\tilde{\mathbf{a}}_i - \tilde{\mathbf{b}}_i] + \mathbf{R}_P^S\mathbf{Q}_{k,i}\begin{bmatrix}\mathbf{j}_{ha,i} \\ \mathbf{j}_{hb,i}\end{bmatrix} & \mathbf{0}_{3\times 2}\end{bmatrix} \quad (23)$$

The velocity of point N is

$$\mathbf{v}_n = \mathbf{v} - (\mathbf{r}+\mathbf{t})\times\boldsymbol{\omega} + \mathbf{R}_P^S\mathbf{K}_n[\dot{\theta}_{ta} \quad \dot{\theta}_{tb}]^T \quad (24)$$

where

$$\mathbf{K}_n = l_t\begin{bmatrix} 0 & cos\theta_{tb} \\ sin\theta_{ta}cos\theta_{tb} & cos\theta_{ta}sin\theta_{tb} \\ -cos\theta_{ta}cos\theta_{tb} & sin\theta_{ta}sin\theta_{tb} \end{bmatrix} \quad (25)$$

And the Jacobian matrix of point N is

$$\mathbf{J}_t = [\mathbf{I} \quad -\tilde{\mathbf{r}} - \tilde{\mathbf{t}} \quad \mathbf{R}_P^S\mathbf{K}_n] \quad (26)$$

2.4 Actuation Jacobian Matrices

To apply the virtual work, we need to express all the virtual displacements with respect to the generalized coordinates, which is carried out in the last two subsections by calculating the Joint Jacobian Matrices and the Point Jacobian Matrices. This subsection, instead, will use these matrices to derive the so-called Actuation Jacobian Matrices which are used to express the virtual displacements of the actuators with respect to the generalized coordinates. Since the actuators are placed on the revolute joints (each universal joint is equivalent to two revolute joints), the actuation Jacobian matrices can be obtained directly from the joint Jacobian matrices.

Note that our quadruped has only eight degrees of freedom while we placed $3\times 4+2=14$ actuators. The six redundant actuators are used to overcome singularities and improve controllability. To distinguish these two different groups of actuation forces, we use $\boldsymbol{\tau}_a$ to represent the eight non-redundant forces and $\boldsymbol{\tau}_p$ to represent the six redundant forces. For convenience, $\mathbf{q}_a = [\theta_{ha,1}, \theta_{ha,2}, \theta_{k,1}, \theta_{k,2}, \theta_{k,3}, \theta_{k,4}, \theta_{ta}, \theta_{tb}]^T$ is chosen as the non-redundant actuation joints. $\mathbf{J}_{\tau,a}$ and $\mathbf{J}_{\tau,p}$ are the corresponding actuation Jacobian matrices. Thus the non-redundant actuation Jacobian matrix can be obtained straightforwardly

$$\mathbf{J}_{\tau,a} = \begin{bmatrix} \mathbf{j}_{ha,1} & 0 & 0 \\ \mathbf{j}_{ha,2} & 0 & 0 \\ \mathbf{j}_{k,1} & 0 & 0 \\ \mathbf{j}_{k,2} & 0 & 0 \\ \mathbf{j}_{k,3} & 0 & 0 \\ \mathbf{j}_{k,4} & 0 & 0 \\ 0\cdots 0 & 1 & 0 \\ 0\cdots 0 & 0 & 1 \end{bmatrix} \quad (27)$$

as well as the redundant actuation Jacobian matrix ($\mathbf{q}_p = [\theta_{ha,3}, \theta_{ha,4}, \theta_{hb,1}, \theta_{hb,2}, \theta_{hb,3}, \theta_{hb,4}]^T$)

$$\mathbf{J}_{\tau,p} = \begin{bmatrix} \mathbf{j}_{ha,3} & 0 & 0 \\ \mathbf{j}_{ha,4} & 0 & 0 \\ \mathbf{j}_{hb,1} & 0 & 0 \\ \mathbf{j}_{hb,2} & 0 & 0 \\ \mathbf{j}_{hb,3} & 0 & 0 \\ \mathbf{j}_{hb,4} & 0 & 0 \end{bmatrix} \quad (28)$$

2.5 Acceleration Analysis

Acceleration can be computed by differentiating the velocity vector. Therefore, differentiating Eq. (19) yields the acceleration of point H_i

$$\mathbf{a}_{h,i} = \dot{\mathbf{v}} + \tilde{\dot{\boldsymbol{\omega}}}\mathbf{a}_i + \tilde{\boldsymbol{\omega}}^2\mathbf{a}_i \quad (29)$$

The acceleration of point K_i can be obtained by differentiating Eq. (21)

$$\mathbf{a}_{k,i} = \dot{\mathbf{v}} + (\tilde{\dot{\boldsymbol{\omega}}} + \tilde{\boldsymbol{\omega}}^2)(\mathbf{a}_i + \mathbf{b}_i) + 2\tilde{\boldsymbol{\omega}}\mathbf{R}_P^S\mathbf{Q}_{k,i}\begin{bmatrix}\dot{\theta}_{ha,i} \\ \dot{\theta}_{hb,i}\end{bmatrix} + \mathbf{R}_P^S\left(\mathbf{Q}_{k,i}\begin{bmatrix}\ddot{\theta}_{ha,i} \\ \ddot{\theta}_{hb,i}\end{bmatrix} + \dot{\mathbf{Q}}_{k,i}\begin{bmatrix}\dot{\theta}_{ha,i} \\ \dot{\theta}_{hb,i}\end{bmatrix}\right) \quad (30)$$

Similarly, the acceleration of point N is obtained from Eq. (24)

$$\mathbf{a}_n = \dot{\mathbf{v}} + (\tilde{\dot{\boldsymbol{\omega}}} + \tilde{\boldsymbol{\omega}}^2)(\mathbf{r}+\mathbf{t}) + 2\tilde{\boldsymbol{\omega}}\mathbf{R}_P^S\mathbf{K}_t\begin{bmatrix}\dot{\theta}_{ta} \\ \dot{\theta}_{tb}\end{bmatrix} + \mathbf{R}_P^S\left(\mathbf{K}_t\begin{bmatrix}\ddot{\theta}_{ta} \\ \ddot{\theta}_{tb}\end{bmatrix} + \dot{\mathbf{K}}_t\begin{bmatrix}\dot{\theta}_{ta} \\ \dot{\theta}_{tb}\end{bmatrix}\right) \quad (31)$$

Note that $\dot{\mathbf{Q}}_{k,i}$ and $\dot{\mathbf{K}}_t$ are the derivatives of $\mathbf{Q}_{k,i}$ and $\mathbf{K}_t$, respectively, which can be directly obtained from Eq. (22) and Eq. (25), respectively.

3 FORMULATION OF THE QUADRUPED DYNAMICS USING VIRTUAL WORK PRINCIPLE

Based on the virtual work principle, for an N rigid body system, the equations of motion can be stated as

$$\sum_{i=1}^{N}\left[\mathbf{J}_{x,i}^T(\mathbf{F}_i - m\dot{\mathbf{v}}_i) + \mathbf{J}_{\omega,i}^T(\mathbf{M}_i - \mathbf{I}_i\dot{\boldsymbol{\omega}}_i - \tilde{\boldsymbol{\omega}}_i\mathbf{I}_i\boldsymbol{\omega}_i)\right] = 0 \quad (32)$$

where $\mathbf{F}_i$ and $\mathbf{M}_i$ are the active force on body i. Moreover, $\mathbf{J}_{x,i}$, $\mathbf{J}_{\omega,i}$ are the Jacobian matrix for the virtual linear and angular displacement respectively. $\mathbf{I}_i$ is the inertia matrix for body i and $\mathbf{v}_i$, $\boldsymbol{\omega}_i$ are the linear and angular velocity. All these terms are measured in the inertial frame.

Although Eq. (32) expresses the full dynamics (both inverse and forward) of the multibody system, the forward dynamics is usually very hard to compute due to the complexity of the Jacobian matrices and Coriolis terms. A practical way to compute the forward dynamics is to use numerical methods (such as Matlab/Simulink) to solve Eq. (32). In this section, the inverse model is formulated first, thanks to its straightforward relationship with Eq. (32). After obtaining the inverse dynamics, the forward model can be obtained directly from the inverse model by extracting the corresponding terms.

3.1 The Virtual Work for Leg i

Since each thigh, shank and the tail in the quadruped are regarded as ideal bars (evenly distributed line mass), before we go into the quadruped dynamics, we would like to derive the virtual work for each leg first by using the technique introduced in [17]. That is, the virtual work of an ideal bar can be lumped to its two endpoints. Figure 4 shows the schematic diagram of a leg and an ideal bar with their kinematic parameters.

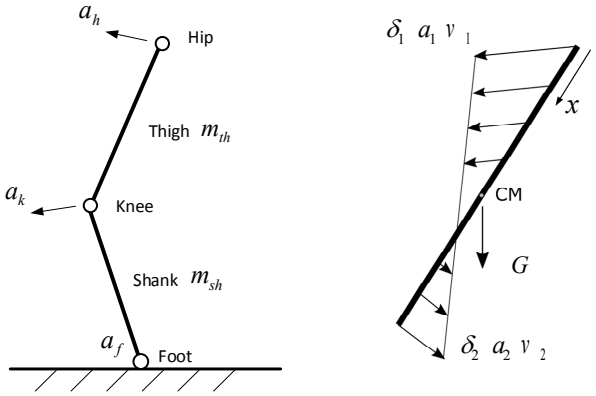

Figure 4. One leg with its foot touching the ground (left) and the velocity distribution on a rigid bar (right)

Therefore, for an ideal bar as shown in Fig. 4, the virtual work due to the inertia force can be computed by the integral

$$\delta W = \int_0^L \boldsymbol{\delta}(x)\mathbf{a}(x)\frac{m}{L}dx \tag{33}$$

in which

$$\boldsymbol{\delta}(x) = \left(1-\frac{x}{L}\right)\boldsymbol{\delta}_1 + \frac{x}{L}\boldsymbol{\delta}_2$$
$$\boldsymbol{a}(x) = \left(1-\frac{x}{L}\right)\mathbf{a}_1 + \frac{x}{L}\mathbf{a}_2$$

are the linear interpolations of the virtual displacement $\boldsymbol{\delta}_1$ $\boldsymbol{\delta}_2$ and the acceleration $\mathbf{a}_1$ $\mathbf{a}_2$ along the bar, respectively. L is the length of the bar. Evaluating Eq. (33) directly yields

$$\delta W = \delta\mathbf{q}^T\frac{m}{3}\left(\mathbf{J}_1^T\mathbf{a}_1 + \mathbf{J}_2^T\mathbf{a}_2 + \frac{\mathbf{J}_2^T\mathbf{a}_1 + \mathbf{J}_1^T\mathbf{a}_2}{2}\right) \tag{34}$$

where $\mathbf{q}$ is the generalized coordinates vector, $\mathbf{J}_1$ and $\mathbf{J}_2$ are the corresponding Jacobian matrices for the two endpoints (This implies $\boldsymbol{\delta}_1 = \mathbf{J}_1\delta\mathbf{q}$ and $\boldsymbol{\delta}_2 = \mathbf{J}_2\delta\mathbf{q}$). Applying this formula on the thigh and shank of each leg gives the virtual work $\delta W_{i,Iner}$ of leg i due to inertia force

$$\delta W_{i,Iner} = \delta\mathbf{q}^T[\mathbf{J}_{k,i}^T\left(\frac{m_{th}+m_{sh}}{3}\mathbf{a}_{k,i} + \frac{m_{th}}{6}\mathbf{a}_{h,i}\right) + \mathbf{J}_{h,i}^T\left(\frac{m_{th}}{3}\mathbf{a}_{h,i} + \frac{m_{th}}{6}\mathbf{a}_{k,i}\right)] \tag{35}$$

where m_{th} and m_{sh} are the mass of the thigh and shank respectively.

The virtual work $\delta W_{i,G}$ due to gravity is obtained as

$$\delta W_{i,G} = \delta\mathbf{q}^T\left(\mathbf{J}_{k,i}^T\frac{m_{th}+m_{sh}}{2} + \mathbf{J}_{h,i}^T\frac{m_{sh}}{2}\right)[0,0,-g]^T \tag{36}$$

3.2 Formulating the Inverse Dynamics of the Quadruped

The inverse dynamics involved finding the actuation force given the motion trajectory of the quadruped. Therefore, the inverse dynamics can be obtained directly from Eq. (32)

$$\mathbf{J}_{\tau,a}^T\boldsymbol{\tau}_a + \mathbf{J}_{\tau,p}^T\boldsymbol{\tau}_p = \boldsymbol{\tau}_b + \boldsymbol{\tau}_t + \sum_{i=1}^{4}\boldsymbol{\tau}_{l,i} \tag{37}$$

where $\boldsymbol{\tau}_a$ is the non-redundant actuation torque, $\boldsymbol{\tau}_p$ is the redundant actuation torque, $\boldsymbol{\tau}_b$, $\boldsymbol{\tau}_t$ and $\boldsymbol{\tau}_{l,i}$ are the generalized torques contributed by the travelling plate, the tail and the ith leg, respectively. $\mathbf{J}_{\tau,a}^T$ and $\mathbf{J}_{\tau,p}^T$ are the corresponding Jacobian matrices for the actuation torque. Their detailed expressions are

$$\boldsymbol{\tau}_b = \mathbf{J}_{b,x}^T m_b(\dot{\mathbf{v}} - [0,0,-g]^T) + \mathbf{J}_{b,\omega}^T(\mathbf{I}_b\dot{\boldsymbol{\omega}} + \tilde{\boldsymbol{\omega}}\mathbf{I}_b\boldsymbol{\omega}) \tag{38}$$

$$\boldsymbol{\tau}_t = \mathbf{J}_t^T m_t(\mathbf{a}_n - [0,0,-g]^T) \tag{39}$$

$$\boldsymbol{\tau}_{l,i} = \mathbf{J}_{k,i}^T\left(\frac{m_{th}+m_{sh}}{3}\mathbf{a}_{k,i} + \frac{m_{th}}{6}\mathbf{a}_{h,i} - \frac{m_{th}+m_{sh}}{2}\begin{bmatrix}0\\0\\-g\end{bmatrix}\right) + \mathbf{J}_{h,i}^T\left(\frac{m_{th}}{3}\mathbf{a}_{h,i} + \frac{m_{th}}{6}\mathbf{a}_{k,i} - \frac{m_{th}}{2}\begin{bmatrix}0\\0\\-g\end{bmatrix}\right) \tag{40}$$

in which m_b and m_t are the mass of the travelling plate and the tail respectively, g is the gravity constant. Note that $\boldsymbol{\tau}_{l,i}$ is obtained by extracting the coefficient terms of $\delta\mathbf{q}$ from the summation of Eq. (35) and Eq. (36). Moreover, the inertia matrix of the travelling plate $\mathbf{I}_b$, matrices $\mathbf{J}_{b,x}$ and $\mathbf{J}_{b,\omega}$ have the form

$$\mathbf{I}_b = \mathbf{R}_P^S\mathbf{I}_b^{(P)}\mathbf{R}_S^P$$
$$\mathbf{J}_{b,x} = [\mathbf{I}_{3\times3} \quad \mathbf{0}_{3\times5}]$$
$$\mathbf{J}_{b,\omega} = [\mathbf{0}_{3\times3} \quad \mathbf{I}_{3\times3} \quad \mathbf{0}_{3\times2}]$$

Note that Eq. (37) has fourteen actuation forces ($\boldsymbol{\tau}_a$ has eight components and $\boldsymbol{\tau}_p$ has six) while the dimension of the equation is only eight (corresponds to the eight independent generalized coordinates). This means that if we regard all actuation forces as unknowns, then there exist infinitely many solutions. Indeed, this is an intrinsic characteristic for an over-actuated system in which there are infinitely many choices of actuation forces to generate the same motion. Therefore, to solve Eq. (37) properly, additional information is required, such as the profile of $\boldsymbol{\tau}_p$. The simplest $\boldsymbol{\tau}_p$ profile is $\boldsymbol{\tau}_p = 0$. This simplifies Eq. (37) to

$$\boldsymbol{\tau}_a = \mathbf{J}_{\tau,a}^{T\ -1}\left(\boldsymbol{\tau}_b + \boldsymbol{\tau}_t + \sum_{i=1}^{4}\boldsymbol{\tau}_{l,i}\right) \tag{41}$$

Obviously, the prerequisite for this solution is that $\mathbf{J}_{\tau,a}$ is nonsingular.

3.3 Formulating the Forward Dynamics from the Inverse Dynamics

In Eq. (37), if we can express all the kinematic terms (acceleration, velocity, and position) with generalized

coordinates, then the inverse dynamics can be written in the forward dynamics form as,

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) = \mathbf{F} \quad (42)$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is the Coriolis term, $\mathbf{G}(\mathbf{q})$ is the gravity and $\mathbf{F}$ is the actuation force. To solve Eq. (42) numerically, the following transformation is required

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}\big(\mathbf{F} - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{G}(\mathbf{q})\big) \quad (43)$$

This form allows numerical integration of $\mathbf{q}$. Therefore, to formulate the forward dynamics, we can just collect all the acceleration terms in Eq. (37), and express them by generalized coordinates. This will give us, both the mass matrix and the Coriolis terms. From the formulation of the inverse dynamics, the following terms can be obtained

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}_b + \mathbf{C}_t + \sum_{i=1}^{4} \mathbf{C}_{l,i} \quad (44)$$

$$\mathbf{C}_b = \mathbf{J}_{b,\omega}^T \tilde{\boldsymbol{\omega}} \mathbf{I}_b \boldsymbol{\omega} \quad (45)$$

$$\mathbf{F} = \mathbf{J}_{\tau,a}^T \boldsymbol{\tau}_a + \mathbf{J}_{\tau,p}^T \boldsymbol{\tau}_p \quad (46)$$

$$\mathbf{G}(\mathbf{q}) = [m_b \mathbf{J}_{b,x}^T + m_t \mathbf{J}_t^T + \sum_{i=1}^{4} \left(\mathbf{J}_{k,i}^T \frac{m_{th,i} + m_{sh,i}}{2} + \mathbf{J}_{h,i}^T \frac{m_{th,i}}{2} \right)] \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \quad (47)$$

Coriolis effect of the tail $\mathbf{C}_t$ can be extracted from Eq. (31) as

$$\mathbf{C}_t = m_t \mathbf{J}_t^T \left(\tilde{\boldsymbol{\omega}}^2 (\mathbf{r} + \mathbf{t}) + 2\tilde{\boldsymbol{\omega}} \mathbf{R}_P^S \mathbf{K}_t \begin{bmatrix} \dot{\theta}_{ta} \\ \dot{\theta}_{tb} \end{bmatrix} + \mathbf{R}_P^S \dot{\mathbf{K}}_t \begin{bmatrix} \dot{\theta}_{ta} \\ \dot{\theta}_{tb} \end{bmatrix} \right) \quad (48)$$

As for the Coriolis forces for the legs, although more complicated, they can be collected from $\mathbf{a}_{k,i}$, and $\mathbf{a}_{h,i}$. The detailed expressions are omitted here due to space limitations.

The mass matrix $\mathbf{M}$ can be obtained by collecting all the second order derivative terms in Eq. (37). However, due to the ideal bar assumption, a better way of deriving the mass matrix for a bar, as defined in Fig.4, is by integrating the kinetic energy T as

$$T = \int_0^L \frac{1}{2} \boldsymbol{v}(x)^2 dm = \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{M} \dot{\mathbf{q}} \quad (49)$$

in which $dm = m_i dx / L$ and

$$\boldsymbol{v}(x) = \left(1 - \frac{x}{L}\right) \boldsymbol{v}_1 + \frac{x}{L} \boldsymbol{v}_2 \quad (50)$$

where, $\boldsymbol{v}(x)$ is the linear interpolation of the velocity along the bar. This yields the mass matrix $\mathbf{M}$

$$\mathbf{M} = \frac{m_i}{3} (\mathbf{J}_1^T \mathbf{J}_1 + \mathbf{J}_2^T \mathbf{J}_2 + \mathbf{J}_1^T \mathbf{J}_2) \quad (51)$$

Note that $\mathbf{M}$ is symmetric due to $\boldsymbol{v}_1^T \boldsymbol{v}_2 = \boldsymbol{v}_2^T \boldsymbol{v}_1$. Therefore, the mass matrix for the quadruped is given by

$$\mathbf{M} = \mathbf{M}_b + \mathbf{M}_t + \sum_{i=1}^{4} \mathbf{M}_{l,i} \quad (52)$$

in which

$$\mathbf{M}_{l,i} = \frac{m_{th}}{3} \left(\mathbf{J}_{h,i}^T \mathbf{J}_{h,i} + \mathbf{J}_{k,i}^T \mathbf{J}_{k,i} + \mathbf{J}_{h,i}^T \mathbf{J}_{k,i} \right) + \frac{m_{sh}}{3} \mathbf{J}_{k,i}^T \mathbf{J}_{k,i} \quad (53)$$

$$\mathbf{M}_b = \begin{pmatrix} m_b \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times2} \\ \mathbf{0}_{3\times3} & \mathbf{I}_b & \mathbf{0}_{3\times2} \\ \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times2} \end{pmatrix} \quad (54)$$

$$\mathbf{M}_t = m_t \mathbf{J}_t^T [\mathbf{I}_{3\times3} \quad -\tilde{\mathbf{r}} - \tilde{\mathbf{t}} \quad \mathbf{R}_P^S \mathbf{K}_t] \quad (55)$$

4 MODEL VERIFICATION

To verify the dynamic model, Matlab/Simulink was used to implement the formulated dynamic models. MSC Adams is also used to do cross-validation. Table 1 lists all the parameters used for simulation. Figure 5 shows the 3D model used in Adams. The feet positions are chosen arbitrarily as

$$[\mathbf{d}_1 \quad \mathbf{d}_2 \quad \mathbf{d}_3 \quad \mathbf{d}_4] = \begin{bmatrix} 0.41 & -0.01 & -0.01 & 0.41 \\ 0.63 & 0.57 & 0.03 & -0.03 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The initial conditions are

$$\begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} [0.2, 0.3, 0.4, 0, 0, 0, 0, 0]^T \\ [0, 0, 0, 0, 0, 0, 0, 0]^T \end{bmatrix}$$

Table 1. Parameter Values for the Quadruped

Parameter = Value	Parameter= Value
$l_{sh} = 0.3\text{m}$	$m_{sh} = 1.2289\text{kg}$
$l_{th} = 0.3\text{m}$	$m_{th} = 0.9453\text{kg}$
$l_t = 0.6\text{m}$	$m_b = 26.9078\text{kg}$
$\|\mathbf{a}_1 - \mathbf{a}_2\| = 0.48\text{m}$	$m_t = 1.4347\text{kg}$
$\|\mathbf{a}_2 - \mathbf{a}_3\| = 0.54\text{m}$	$\mathbf{I}_b^{(U)} = \text{diag}([0.8404, 0.3793, 1.2125])\text{kgm2}$
$\|\mathbf{r}\| = 0.34\text{m}$	$g = 9.8067\text{m/s}^2$

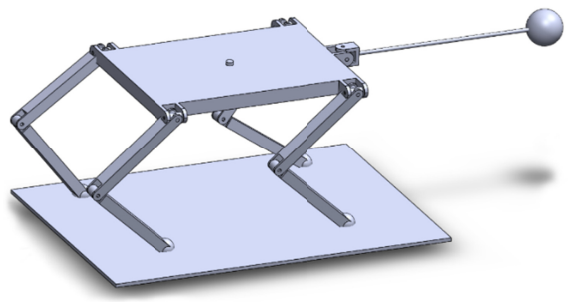

Figure 5. 3D model of the quadruped for simulation

4.1 Verification of the forward model

We first verify the forward dynamic model. This is done by comparing the motions computed by Adams and Simulink separately under the same actuation torque. The actuation torque profile is given by,

$$\boldsymbol{\tau}_a = [0, 0, 15, 15, 20.5, 20.5, -7.7, -10sin(31.4t)]^T$$
$$\boldsymbol{\tau}_p = [0, 0, 0, 0, 0, 0]^T$$

This torque profile drives the quadruped to achieve a yaw maneuvering behavior. Fig. 6 is the motion comparison between Adams and Simulink.

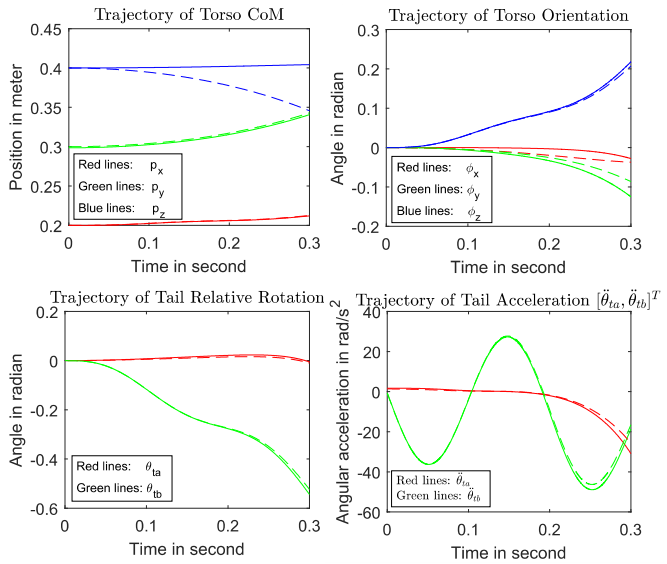

Figure 6. Comparison of simulated motion by Adams (solid lines) and Simulink (dashed lines)

From Fig. 6, the quadruped motion computed by our model does match the simulation results, as generated by Adams. However, the errors between these two approaches are also noticeable, especially for the z component of the torso CM position $\mathbf{p}$. These errors may be the result of two aspects: (1) numerical error, (2) model error. The numerical error includes the error drift of the integrator, the accuracy loss while computing $\mathbf{M}^{-1}$ for Eq. (43), and the error induced by the DAE/ODE solver (Adams uses GSTIFF-I3 solver with a tolerance of 1E-3 while Simulink uses fixed step ODE4 solver. Both implement a time step length of 0.0001s). The model error is due to the assumptions that we considered each thigh and shank as an ideal bar and the geometric center of the torso as its mass center, which is not valid in Adams.

4.2 Verification of the inverse model

The forward model can be used to verify the inverse model. That is, using the inverse model, we can calculate the required torque and input this torque into the forward model. Theoretically, if the inverse model is correct, the forward model should generate the same desired motion. However, due to the error drift of the numerical integrator (referring to Fig. 6 and Fig. 8), the simulated motion deviates from the desired motion after a short period of time (less than 0.2s). Figure 8 shows the deviation of the simulated motion (circled line) from the desired motion (solid line).

Another important source for the deviation might come from the inverse model itself. In the simulation, Eq. (41) is used to compute the non-redundant actuation forces $\boldsymbol{\tau}_a$, for which $\mathbf{J}_{\tau,a}$ has to be invertible. However, in practice, letting $\mathbf{J}_{\tau,a}$ be invertible is not enough. The numerical error of computing $\mathbf{J}_{\tau,a}{}^{-1}$ increases dramatically as $\mathbf{J}_{\tau,a}$ gets closer to its singularities. Since $\mathbf{J}_{\tau,a}$ consists of $\mathbf{j}_{ha,1}$ $\mathbf{j}_{ha,2}$, $\mathbf{j}_{k,1}$, $\mathbf{j}_{k,2}$, $\mathbf{j}_{k,3}$, $\mathbf{j}_{k,4}$, $\mathbf{j}_{ta}$ and $\mathbf{j}_{tb}$, so any two of these being the same will lead to $\mathbf{J}_{\tau,a}$ being singular. In our simulation, we observed that $\mathbf{j}_{ha,1}$ and $\mathbf{j}_{ha,2}$ were very close at multiple points. This verifies the error

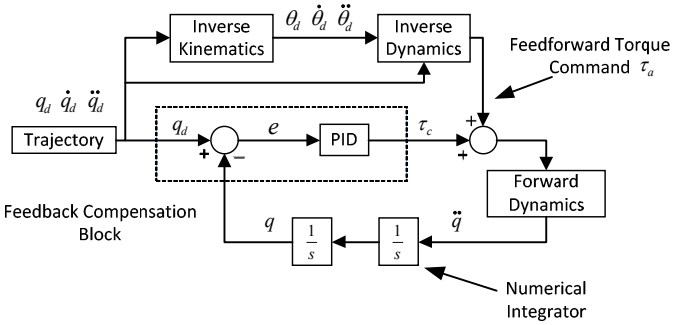

Figure 7. The feed forward control scheme used in the inverse model verification

due to the numerical inversion of $\mathbf{J}_{\tau,a}$.

Therefore, to eliminate these errors, a feedback controller (shown in Fig. 7) is applied to the system, to observe the trajectory of the compensation torque given by the feedback block. The results are shown in Fig. 8 and Fig. 9, which show that the compensation torque asymptotically approaches zero. This means that the actuation torque computed by the inverse model does generate the desired motion after the numerical error is under control. Figure 8 also shows the resulting simulated trajectory by using the controller. For this simulation, the desired motion is a pitch motion is given by

$$q = \left[0.2, 0.3, 0.4, \frac{\pi}{36} sin\left(\frac{2\pi}{0.3}t\right), 0, 0, 0, \frac{\pi}{18} sin\left(\frac{2\pi}{0.3}t + \pi\right)\right]^T$$

and the redundant actuation force is $\tau_p = [0, 0, 0, 0, 0, 0]^T$.

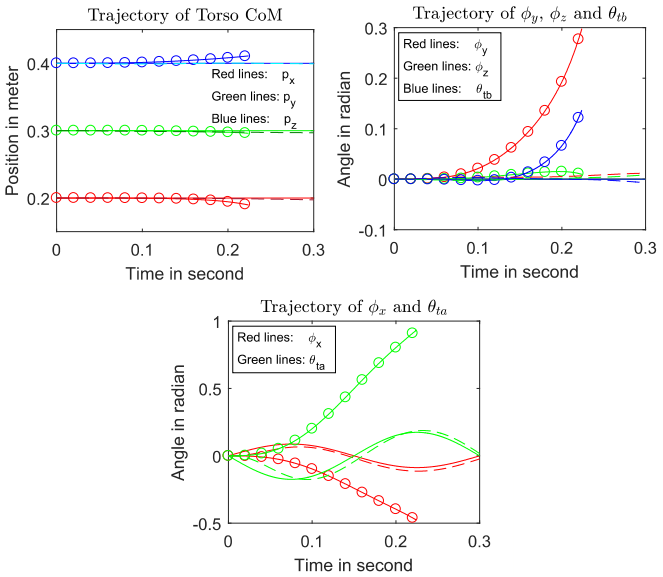

Figure 8. Simulated quadruped behaviors with control (dashed lines) and without control (circled lines). Solid lines show the desired trajectories.

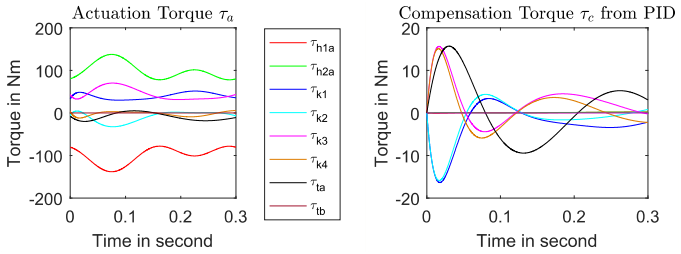

Figure 9. The required torque and the compensation torque τ_c to generate the desired motion

5 CONCLUSION

By applying the virtual work principle, this paper presented a methodology to formulate the forward and inverse dynamical models for a general quadruped with a robotic tail. This model will be used to analyze the influence of a tail on the motion of the quadruped. However, the same model can also be used as an alternative for the Lagrangian and Newton-Euler based models. The formulation process follows a standard procedure of modeling multibody dynamics. At first, the kinematic analysis is conducted to obtain all necessary Jacobian matrices and accelerations, then, these terms are substituted in the equation of motion formulated by the virtual work principle. Lastly, these terms are rearranged to derive the inverse and forward dynamical models. This paper also performed a numerical experiment which showed that the forward dynamical model did predict the same motion as simulated in Adams. The inverse dynamical model did generate the required actuation torque used in the forward dynamics simulation.

Future work will focus on understanding the influences of a tail on the motion of a quadruped as well as on modeling the hybrid system for locomotion. Eventually, a quadruped with a tail will be used to test the maneuverability and stabilization of the robot during fast motions. In addition, future work will be focused on implementing a serpentine robotic tail instead of a pendulum-like structure, due to its ability to provide better maneuverability.

ACKNOWLEDGMENTS

This research is based upon work supported by the National Science Foundation under Grant No. 1557312.

REFERENCES

[1] Heim, S.W., Ajallooeian, M., Eckert, P., Vespignani, M. and Ijspeert, A.J., 2016. On designing an active tail for legged robots: simplifying control via decoupling of control objectives. Industrial Robot: An International Journal, 43(3), pp.338-346.

[2] Jusufi, A., Kawano, D.T., Libby, T. and Full, R.J., 2010. Righting and turning in mid-air using appendage inertia: reptile tails, analytical models and bio-inspired robots. Bioinspiration & Biomimetics, 5(4), p.045001.

[3] Rone, W. and Ben-Tzvi, P., 2016. Dynamic Modeling and Simulation of a Yaw-Angle Quadruped Maneuvering With a Planar Robotic Tail. Journal of Dynamic Systems, Measurement, and Control, 138(8), p.084502.

[4] Patel, A. and Braae, M., 2014, May. Rapid acceleration and braking: Inspirations from the cheetah's tail. 2014 ICRA, pp. 793-799.

[5] Patel, A. and Boje, E., 2015. On the conical motion of a two-degree-of-freedom tail inspired by the Cheetah. IEEE Transactions on Robotics, 31(6), pp.1555-1560.

[6] Briggs, R., Lee, J., Haberland, M. and Kim, S., 2012, October. Tails in biomimetic design: Analysis, simulation, and experiment. 2012 IROS, pp. 1473-1480.

[7] Rone, W.S. and Ben-Tzvi, P., 2014. Mechanics modeling of multisegment rod-driven continuum robots. Journal of Mechanisms and Robotics, 6(4), p.041006.

[8] Rone, W.S., 2017, August. Maneuvering and stabilizing control of a quadrupedal robot using a serpentine robotic tail. IEEE-CCTA 2017 pp. 1763-1768.

[9] Patel, A. and Braae, M., 2015. An actuated tail increases rapid acceleration manoeuvres in quadruped robots. In Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering (pp. 69-76). Springer, Cham.

[10] Bennani, M. and Giri, F., 1996. Dynamic modelling of a four-legged robot. Journal of Intelligent & Robotic Systems, 17(4), pp.419-428.

[11] Perrin, B., Chevallereau, C. and Verdier, C., 1997, April. Calculation of the direct dynamic model of walking robots: Comparison between two methods. IEEE-ICRA 1997 (Vol. 2, pp. 1088-1093).

[12] Zamani, A., Khorram, M. and Moosavian, S.A.A., 2011, October. Dynamics and stable gait planning of a quadruped robot. In Control, Automation and Systems (ICCAS), 2011 11th International Conference on (pp. 25-30). IEEE.

[13] Ding, X. and Chen, H., 2016. Dynamic modeling and locomotion control for quadruped robots based on center of inertia on SE (3). Journal of Dynamic Systems, Measurement, and Control, 138(1), p.011004.

[14] Mistry, M., Buchli, J. and Schaal, S., 2010, May. Inverse dynamics control of floating base systems using orthogonal decomposition. IEEE-ICRA 2010 (pp. 3406-3412).

[15] Shah, S.V., Saha, S.K. and Dutt, J.K., 2012. Modular framework for dynamic modeling and analyses of legged robots. Mechanism and Machine Theory, 49, pp.234-255.

[16] Lee, K.P., Koo, T.W. and Yoon, Y.S., 1998, May. Real-time dynamic simulation of quadruped using modified velocity transformation. IEEE-ICRA 1998 (Vol. 2, pp. 1701-1706).

[17] Codourey, A., 1998. Dynamic modeling of parallel robots for computed-torque control implementation. IJRR, 17(12), pp.1325-1336.

[18] Tsai, L.W., 2000. Solving the inverse dynamics of a Stewart-Gough manipulator by the principle of virtual work. Transactions-American Society of Mechanical Engineers Journal of Mechanical Design, 122(1), pp.3-9.