Year 2 Project

Simulating Quadrupedal Locomotion in PyBullet

Minghong Xu (ID 201601082)

Name Surname2 (ID 2052234)

Group 00

Supervised by Dr Murat Uney

March 17, 2022

Abstract

This document serves as a template to show you how to present and structure your project dissertation. In the interests of uniformity of format, you are asked not to significantly alter the format and layout of this LaTeX document. The chapter names and structure are intended as a guide, and you are welcome to change these as appropriate. While this is not a definitive guide to dissertation writing, it is intended as a guide to assist in documenting and presenting your project work in an academic and professional manner, and reflects the expectations of those in academia and industry who are likely to read your report. Appended to this guide are some real examples of common mistakes that should be avoided.

Your Abstract is expected to be between 100 and 250 words in length, and should summarise the problem, outline the approach adopted, and summarise the project findings or results. It's the first (and possibly the only) part to be read, and should provide a snapshot of the whole project in 2 or 3 paragraphs.

Declaration

I confirm that I have read and understood the University's definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Assessment (Appendix L).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Rationale

There are currently three types of mobile robots: wheeled robots, crawler robots, and footed robots. It is difficult to see the difference in the moving efficiency of the three mobile robots on an ideal level of the ground. However, in reality, there are more bumpy and slippery scenarios in which the mobility efficiency of the footed robot is unmatched by the other two robots. Therefore, the application prospect of the footed robot is very broad.

The movement of a footed robot in a complex environment is inseparable from the ability to adjust the posture every moment. This ability helps the robot to maintain balance and complete movement during motion. This project will use the forward and inverse kinematics algorithm to solve the robot joint coordinates before and after posture adjustment, and control the model in the Pybullet simulation environment to complete the simulation of the posture adjustment of the quadruped robot.

1.2 Objectives

The objective of this project is to develop a posture adjustment program for a quadruped robot in three directions in three-dimensional space. The motions completed by the posture adjustment in the three directions are pitch motion, roll motion, and yaw motion. Finally, three programs should be combined to complete the posture adjustment of the quadruped robot in any direction in space.

In addition, the height adjustment of the robot can also be regarded as a part of the posture adjustment. The height adjustment program is combined into the posture adjustment program to achieve the additional goal of adjusting the posture of the quadruped robot at any height.

Chapter 2

Materials and Methods

2.1 Materials

2.1.1 Simulation Enviornment

- **programming language:** Python 3.10
- **robotics simulator:** PyBullet 3.2.0
- **quadruped robot 3D model:** Unitree A1
- **packages that extend Python's array programming capability:**
  Numpy 1.22.2
  Pandas 1.4.0

2.1.2 Development Environment

- **terminal emulator:** Windows Terminal
- **shell:** PowerShell 7.2.1
- **package manager:** Scoop
- **Python version manager:** pyenv-win
- **Python packaging and dependency manager:** Poetry
- **version control system:** Git
- **code hosting platform:** GitHub

- **languages for documentation:**
  - Markdown
  - LaTeX 2_ε
    - TeX distribution: MiKTeX 21.12
    - typesetting engine: XeTeX
    - reference management: BibTeX
  - Mermaid 8.14.0
- **editor:** VSCode
  - **Python support:** Python extension pack 2022.2.1924087327
  - **Markdown support:** Markdown All in One 3.4.0
  - **LaTeX support:** LaTeX Workshop 8.23.0
- **3D CAD modeling:** Shapr3D
- **file format:** EditorConfig
- **blog framework:** Hexo
  - **theme:** NexT
  - **Markdown renderer:** hexo-renderer-marked
  - **MathJax renderer:** hexo-filter-mathjax
- **static site hosting service:** GitHub Pages

2.2 Methods

2.2.1 Modelling

Firstly, a geometric model of the quadruped robot was established.

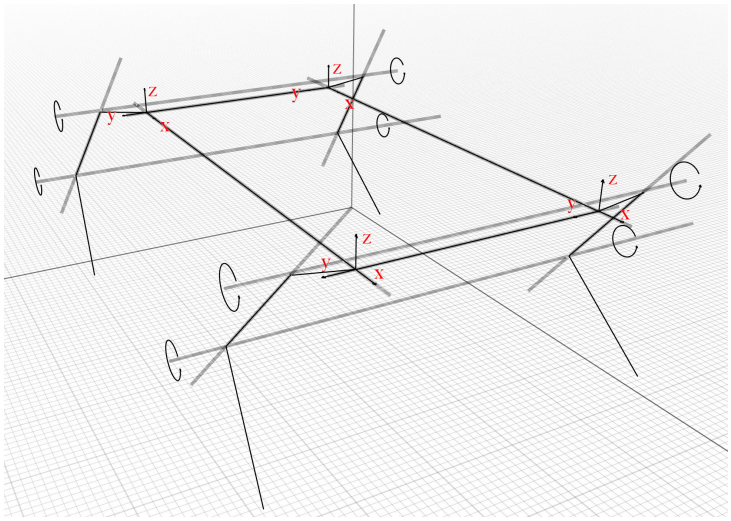


Figure 2.1: Geometric model of the quadruped robot

Figure 2.1 shows a trimetric view of the model. The body is reduced to a square and the legs are reduced to three connecting rods. The joint between the body and the legs is called hip. The uppermost rod of the leg is called the hip offset, followed by the thigh and finally the shank. In a robot entity, the distance from centre of motor controlling hip abduction/adduction to top centre of the thigh is approximated by the hip offset.

The leg has four important parts. There are three movable parts, named hip abduction/adduction (abd/add) joint, hip flextion/extension (fle/ext) joint, and knee joint from top to bottom. The end of the leg is called toe.

In order to describe leg movements using mathematics, it is first necessary to establish a coordinate system. In terms of leg movements, the hip and the toe are the parts of the leg that receive the most attention; because once the position of the toe in relation to the hip has been determined, the shape of the leg is fixed. The hip is located in the body and is a immovable point for the body, so setting it as the origin is convient for thinking. The positive direction of the coordinate system is shown in Figure 2.1. There will be a coordinate system on each hip. This means that the four legs will be studied separately.

Specifying initial position of the motor and positive direction of rotation is as important as establishing the coordinate system. The quadruped robot model used for this project, Unitree A1, specifies them in its unified-robotics-description-format file, so these provisions are directly followed. The initial position of the motors and the positive direction of rotation is shown in Figure ¡++¿.

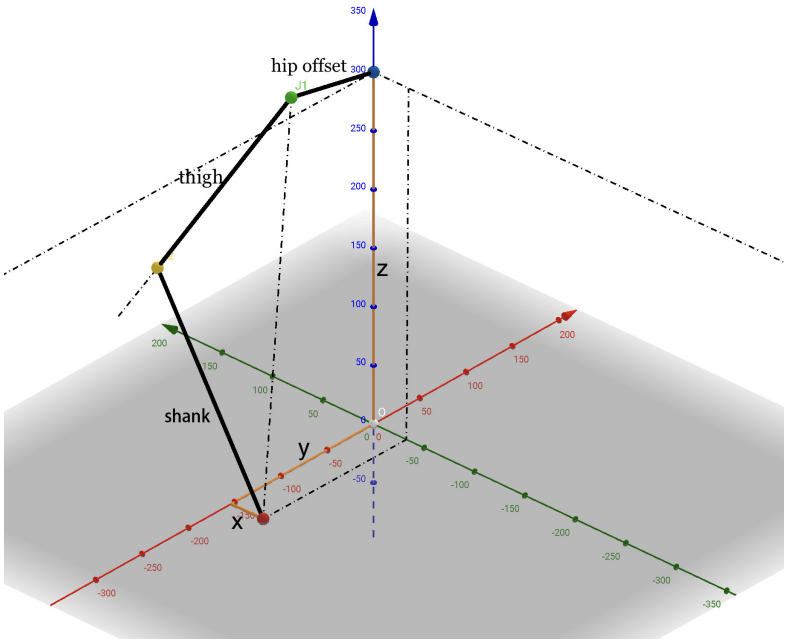


Figure 2.2: Right leg model

The Unitree A1's unified-robotics-description-format file also specifies the boundary positions of the motors. They are listed in Table 2.1. For the sake of readability, front-left is abbreviated as fl, front-right as fr, hind-left as hl, and hind-right as hr.

Table 2.1: Motors boundary positions

Motors	Positive Boundary (rad)	Negative Boundary (rad)
fl hip abd/add	0.803	-0.803
fr hip abd/add	0.803	-0.803
hl hip abd/add	0.803	-0.803
hr hip abd/add	0.803	-0.803
fl hip fle/ext	4.189	-1.047
fr hip fle/ext	4.189	-1.047
hl hip fle/ext	4.189	-1.047
hr hip fle/ext	4.189	-1.047
fl knee	-0.916	-2.697
fr knee	-0.916	-2.697
hl knee	-0.916	-2.697
hr knee	-0.916	-2.697

The next step is to derive the relationship between the toe coordinates before and after posture adjustment. The posture adjustment including pitching, yawing, rolling, and squatting.

Pitching

When pitching, the body length is constant, and the central axis of the body remains in the same position. Using these two conditions, the coordinates after pitching can be obtained by transforming the coordinate system three times. Take the front-right leg as an example:

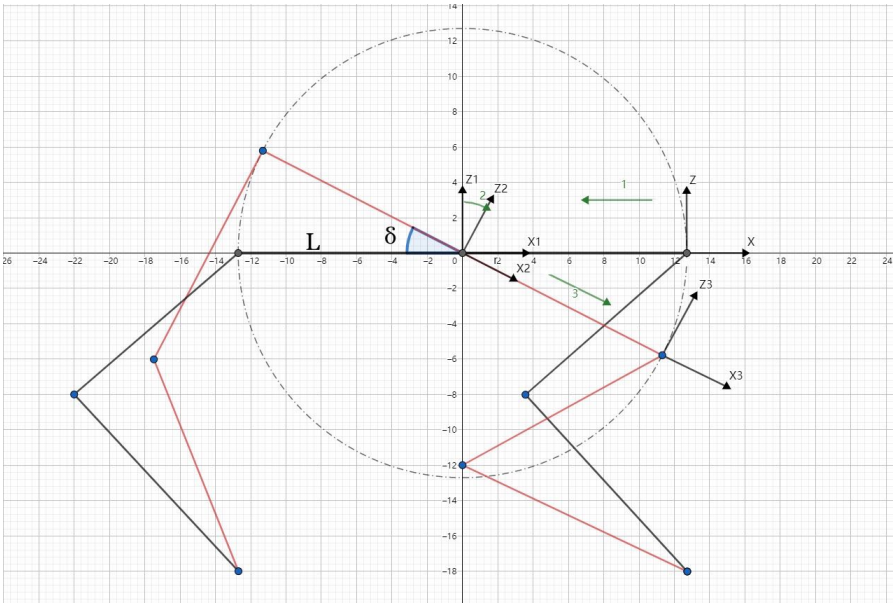


Figure 2.3: Coordinate transformations in pitching

The process of translating the coordinate system from hip to the central axis can be described by pre-multipling a transformation matrix:

$$\begin{bmatrix} x_{\text{after translating}} \\ z_{\text{after translating}} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ z \\ 1 \end{bmatrix} \tag{2.1}$$

where L is half body length.

Then, rotate the coordinate system by the pitching angle δ and transform it to the new location of the hip. The matrix multiplication for this process is

$$\begin{bmatrix} x_{\text{after pitching}} \\ z_{\text{after pitching}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\delta & -\sin\delta & -L \\ \sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\text{after translating}} \\ z_{\text{after translating}} \\ 1 \end{bmatrix} \tag{2.2}$$

The coordinates after pitching are obtained. Note that the y-coordinate does not change when pitching.

The two two matrix multiplication above can be combined into one

$$\begin{bmatrix} x_{\text{after pitching}} \\ z_{\text{after pitching}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\delta & -\sin\delta & L \times \cos\delta - L \\ \sin\delta & \cos\delta & L \times \sin\delta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ z \\ 1 \end{bmatrix} \tag{2.3}$$

The above calculation process is all for the front-right leg. By calculating the other legs it can be seen that for both **front** legs Eq 2.3 holds. For **hind** legs, the equation is

$$\begin{bmatrix} x_{\text{after pitching}} \\ z_{\text{after pitching}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\delta & -\sin\delta & -L \times \cos\delta + L \\ \sin\delta & \cos\delta & -L \times \sin\delta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ z \\ 1 \end{bmatrix} \tag{2.4}$$

Adding the information that y does not change during pitching to the matrix multiplication, the final form for the **front** legs is

$$\begin{bmatrix} x_{\text{after pitching}} \\ y_{\text{after pitching}} \\ z_{\text{after pitching}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\delta & 0 & -\sin\delta & L \times \cos\delta - L \\ 0 & 1 & 0 & 0 \\ \sin\delta & 0 & \cos\delta & L \times \sin\delta \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.5}$$

and for the **hind** legs is

$$\begin{bmatrix} x_{\text{after pitching}} \\ y_{\text{after pitching}} \\ z_{\text{after pitching}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\delta & 0 & -\sin\delta & -L \times \cos\delta + L \\ 0 & 1 & 0 & 0 \\ \sin\delta & 0 & \cos\delta & -L \times \sin\delta \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.6}$$

Yawing

The above idea also applies to yawing. One difference is that the matrix for translating each of the four coordinate systems is different, so that ultimately each leg corresponds to a matrix. See Figure 2.4.

As the derivation has already been described in section Pitching, it is omitted here, and the final form of matrix multiplication for yawing for each leg are listed directly. Give that the yawing angle is β, half body length is L, and half body width is W.

For **front-right** leg:

$$\begin{bmatrix} x_{\text{after yawing}} \\ y_{\text{after yawing}} \\ z_{\text{after yawing}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & L \times \cos\beta - W \times \sin\beta - L \\ \sin\beta & \cos\beta & 0 & L \times \sin\beta + W \times \cos\beta - W \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.7}$$

For **front-left** leg:

$$\begin{bmatrix} x_{\text{after yawing}} \\ y_{\text{after yawing}} \\ z_{\text{after yawing}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & L \times \cos\beta + W \times \sin\beta - L \\ \sin\beta & \cos\beta & 0 & L \times \sin\beta - W \times \cos\beta + W \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.8}$$

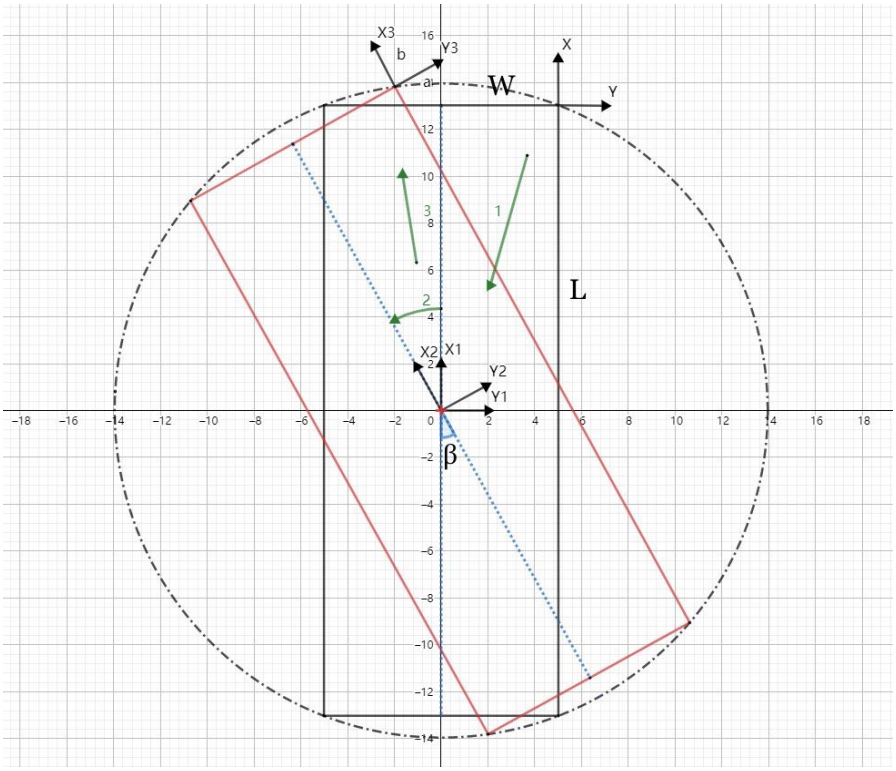


Figure 2.4: Coordinate transformations in yawing

For **hind-right** leg:

$$\begin{bmatrix} x_{\text{after yawing}} \\ y_{\text{after yawing}} \\ z_{\text{after yawing}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & -L\times\cos\beta - W\times\sin\beta + L \\ \sin\beta & \cos\beta & 0 & -L\times\sin\beta + W\times\cos\beta - W \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.9}$$

For **hind-left** leg:

$$\begin{bmatrix} x_{\text{after yawing}} \\ y_{\text{after yawing}} \\ z_{\text{after yawing}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & -L\times\cos\beta + W\times\sin\beta + L \\ \sin\beta & \cos\beta & 0 & -L\times\sin\beta - W\times\cos\beta + W \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.10}$$

Rolling

Unlike yawing, in rolloing, the right two legs share a transformation matrix, as do the left two legs. This is because the translation operation of the coordinate system is different for the left and right sides, while the rotation operation is the same for each leg.

Give that rolling angle is λ, and half body width is W. Again, the derivation is omitted and only the final form of matrix multiplication is presented.

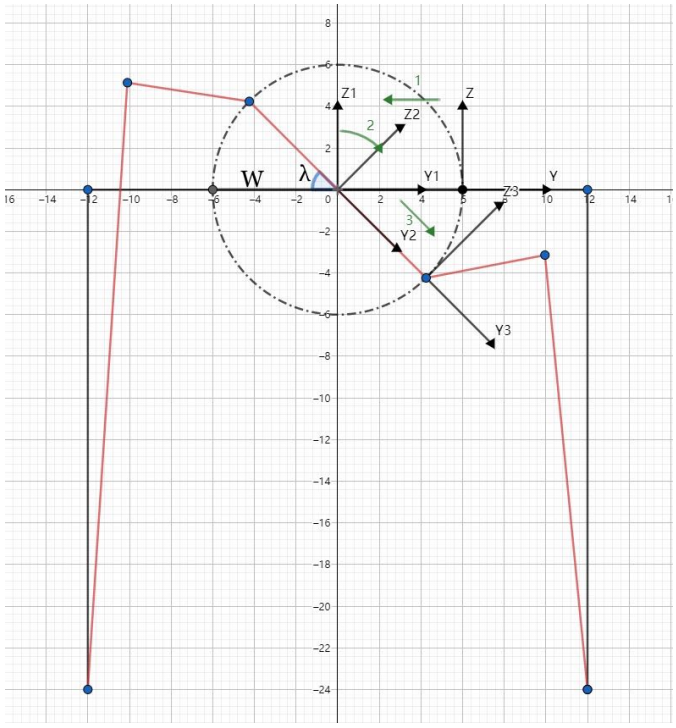


Figure 2.5: Coordinate transformations in rolling

For **right** legs:

$$\begin{bmatrix} x_{\text{after rolling}} \\ y_{\text{after rolling}} \\ z_{\text{after rolling}} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\lambda & -\sin\lambda & W \times \cos\lambda - W \\ 0 & \sin\lambda & \cos\lambda & W \times \sin\lambda \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.11}$$

For **left** legs:

$$\begin{bmatrix} x_{\text{after rolling}} \\ y_{\text{after rolling}} \\ z_{\text{after rolling}} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\lambda & -\sin\lambda & -W \times \cos\lambda + W \\ 0 & \sin\lambda & \cos\lambda & -W \times \sin\lambda \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.12}$$

Squatting

Squatting is actually the process of continuously changing the height of each hip above the ground. In all cases, the z-coordinate represents this height. Assuming that the standing surface does not change, then simply changing the z-coordinate can achieve a squat.

2.2.2 Forward and Inverse Kinematics

In the previous section, the transformation matrix of the coordinates of the toe after posture adjustment was obtained. However, only the motor position can be directly controlled, and also the robot's controller can only return information about the motor position and not the coordinates of the toe relative to the hip joint. Therefore, in order to control the motor position to achieve posture adjustment, both forward and inverse kinematics are required.

The forward kinematics means finding the coordinates of toe relative to the hip given motors position which is servo angle, while inverse kinematics is the opposite. This conversion relation can be derived elegantly using several mathematical concepts such as Euler angle. However, as it is too far beyond the knowledge of the team, the choice here is to use basic geometric knowledge to describe this relationship.

It is necessary to discuss the left and right sides in separate cases, as the Unitree A1's right-side motor configuration of the hip abd/add joint which includes initial position and positive direction does not consistent with the left side. Again, to save space, the equations on both sides are given directly.

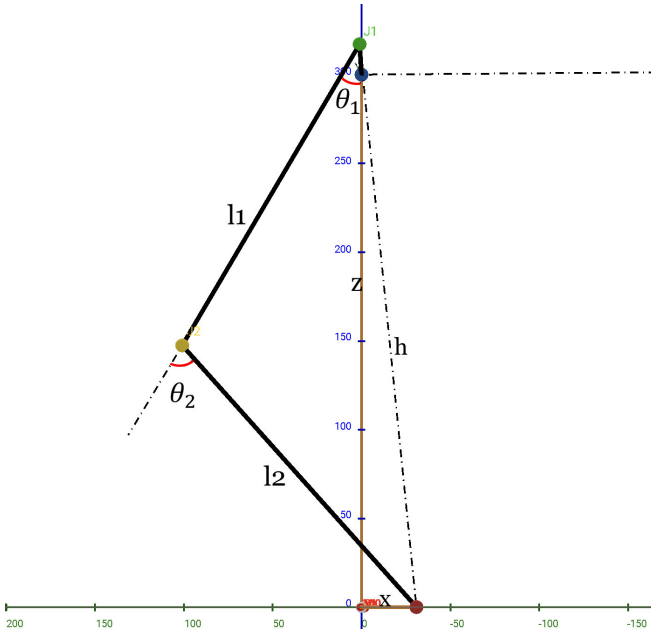


Figure 2.6: Left-side view of the right leg model

Give that thigh length is l_1, shank length is l_2, distance from the toe to the top point of the thigh rod is h, position of the hip fle/ext joint is θ_1, and position of the knee joint is θ_2. The following equation can be obtained easily from the left-side view of the right leg model shown in Figure 2.6.

$\theta_1, \theta_2 \rightarrow x, h$:

$$x = -l_1 \times \sin\theta_1 - l_2 \times \sin(\theta_1 + \theta_2)$$
$$h = l_1 \times \cos\theta_1 + l_2 \times \cos(\theta_1 + \theta_2)$$

$x, h \rightarrow \theta_1, \theta_2$:

$$\cos\theta_2 = \frac{-l_1^2 - l_2^2 + x^2 + h^2}{2l_1 l_2}$$
$$\sin\theta_2 = -\sqrt{1 - \cos^2\theta_2}$$
$$\theta_1 = \arccos\left(\frac{l_1^2 + x^2 + h^2 - l_2^2}{2l_1\sqrt{x^2 + h^2}}\right) - \text{arctan2}\left(\frac{x}{h}\right)$$
$$\theta_2 = \text{arctan2}\left(\frac{\sin\theta_2}{\cos\theta_2}\right)$$

Give that the hip offset is o and position of hip abd/add joint is θ_3. A second set of equations can be obtained from the front view of the right leg model (Figure 2.7). As mentioned above, the motor configuration of hip abd/add joint is different on the left and right side; hence, two sets of equations are required.

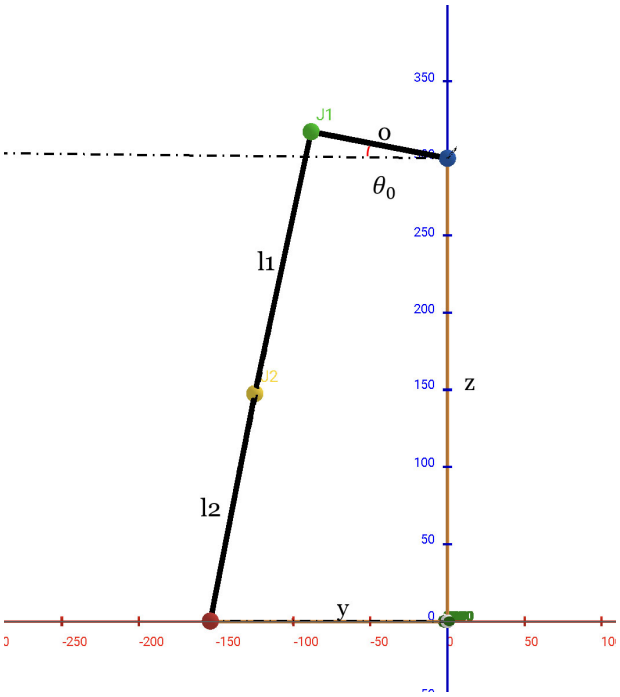


Figure 2.7: Front view of the right leg model

For the **right** side:

$\theta_0 \rightarrow y, z$:

$$y = o \times \cos\theta_0 - h \times \sin\theta_0$$
$$z = -o \times \sin\theta_0 - h \times \cos\theta_0$$

$y, z \to \theta_0$:

$$\theta_0 = \arctan2(\frac{|z|}{y}) - \arctan2(\frac{h}{o})$$

but for the **left** side:

$\theta_0 \to y, z$:

$$y = -o \times \cos\theta_0 - h \times \sin\theta_0$$
$$z = o \times \sin\theta_0 - h \times \cos\theta_0$$

$y, z \to \theta_0$:

$$\theta_0 = \arctan2(\frac{h}{o}) - \arctan2(\frac{|z|}{-y})$$

Summary

Forward kinematics:

$\theta_1, \theta_2 \to x, h$:

$$x = -l_1 \times \sin\theta_1 - l_2 \times \sin(\theta_1 + \theta_2) \tag{2.13}$$
$$h = l_1 \times \cos\theta_1 + l_2 \times \cos(\theta_1 + \theta_2) \tag{2.14}$$

$\theta_0 \to y, z$ for **right** side:

$$y = o \times \cos\theta_0 - h \times \sin\theta_0 \tag{2.15}$$
$$z = -o \times \sin\theta_0 - h \times \cos\theta_0 \tag{2.16}$$

$\theta_0 \to y, z$ for **left** side:

$$y = -o \times \cos\theta_0 - h \times \sin\theta_0 \tag{2.17}$$
$$z = o \times \sin\theta_0 - h \times \cos\theta_0 \tag{2.18}$$

Inverse kinematics:

$x, h \to \theta_1, \theta_2$:

$$\cos\theta_2 = \frac{-l_1^2 - l_2^2 + x^2 + h^2}{2l_1 l_2} \tag{2.19}$$
$$\sin\theta_2 = -\sqrt{1 - \cos^2\theta_2} \tag{2.20}$$
$$\theta_1 = \arccos(\frac{l_1^2 + x^2 + h^2 - l_2^2}{2l_1\sqrt{x^2 + h^2}}) - \arctan2(\frac{x}{h}) \tag{2.21}$$
$$\theta_2 = \arctan2(\frac{\sin\theta_2}{\cos\theta_2}) \tag{2.22}$$

$y, z \rightarrow \theta_0$ for **right** side:

$$\theta_0 = \arctan2(\frac{|z|}{y}) - \arctan2(\frac{h}{o}) \tag{2.23}$$

$y, z \rightarrow \theta_0$ for **left** side:

$$\theta_0 = \arctan2(\frac{h}{o}) - \arctan2(\frac{|z|}{-y}) \tag{2.24}$$

2.2.3 Coding

Equation derivation and coding were carried out simultaneously. After a set of equations for a movement had been derived, a simple Python script was used to verify correctness. If the result seen in the simulator was correct, the algorithm in the script would then be encapsulated into a library which is named liba1.py (See Appendix ¡++¿).

Once a movement had been ecapsulated, a corresponding application was written for debugging and testing purpose. After all the movements completed, a demo application was written which used in presentation of bench inspection. See Appendix ¡++¿.

Chapter 3

Results and Analysis

The results of the project demonstrate the locomotion of quadrupedal robot, which are pitching, yawing, rolling, and squatting. By adjusting parameters through corresponding custom sliders, a series of actions are obtained through screenshots and merged into figures, where certain text interpretation would be executed. Meanwhile, the analysis would contribute to the comparison among states of joints. The motions of the quadrupedal robot would be dissected respectively.

3.1 Squatting

Squatting, for the quadruped robot, is the action that makes the gravity centre of its main body rise or fall vertically. By pulling the custom slider of squatting to adjust the parameters, a series of actions of the quadruped robot when squatting is shown in Fig. 3.1.

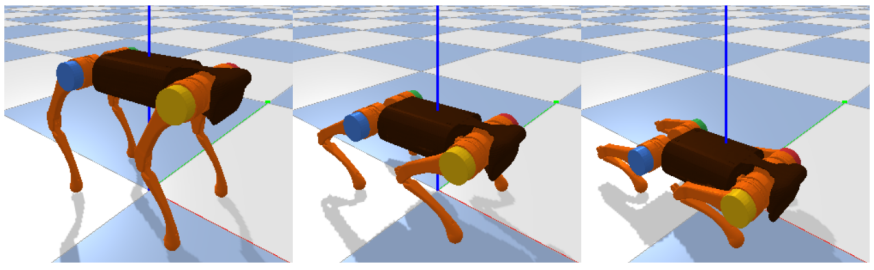


Figure 3.1: Squatting

In the initial position, the distance from hip joints to toe joints is 270. While the slider of debug parameter was pulled to the left, the distance became smaller, the body of the robot descended vertically. While the slider of debug parameter was pulled to the right, the distance became larger, and the robot's body ascended vertically.

The state of abduction/adduction hip joints would retain the initial state while squatting. The state of flection/extension hip joints and knee joints would change corresponding to the distance from the toe. Judging from the simulation results, the robot's legs were crooked while the robot's trunk moves downward, and were extended while the robot's trunk move upward. The state of knee joints. The state of the knee joints determined the current highest position of the robot, and then the actual position of the robot was adjusted by controlling the state of the hip joint.

3.2 Pitching

Pitching, for the quadruped robot, is the action of rotating about the transverse axis [**?**]. By pulling the custom slider of pitching to adjust the parameters, a series of actions of the quadruped robot when pitching is shown in Fig. 3.2.

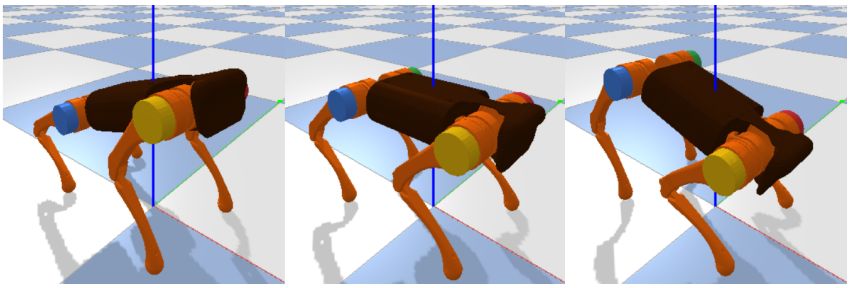


Figure 3.2: Pitching

In the initial position, the body of the quadruped robot remained horizontal, which means that the pitch angle was 0. While the slider of debug parameter was pulled to the left, the pitch angle became negative, and the robot's body made counterclockwise rotation about the transverse axis. While the slider of debug parameter was pulled to the right, the pitch angle became positive, and the robot's body made clockwise rotation about the transverse axis.

As mentioned in the method, the motion of the robot body only occurs in the plane perpendicular to the transverse axis. Therefore, the variations of joints' states between the left and right legs are consistent, unilateral legs need to be evaluated in the analysis. Compared to the motions of yawing and rolling, the motion of pitching is more straightforward, because the state of abduction/adduction hip joints would retain the initial state. However, in the process of pitching, the state of flection/extension hip joints and knee joints would change with the flection or extension of the legs. While the legs were extended, the calves rotated clockwise, from the left perspective, relative to the knee joints., the thighs rotated counterclockwise, from the left perspective, relative to the flection/extension hip joints. While the legs were crooked, the calves rotated counterclockwise, from the left perspective, relative to the knee joints, the thighs rotated clockwise, from the left perspective, relative to the flection/extension hip joints.

3.3 Yawing

Yawing, for the quadruped robot, is the action of rotating about the normal axis [?]. By pulling the custom slider of squatting to adjust the parameters, a series of actions of the quadruped robot when yawing are shown in Fig. 3.3.

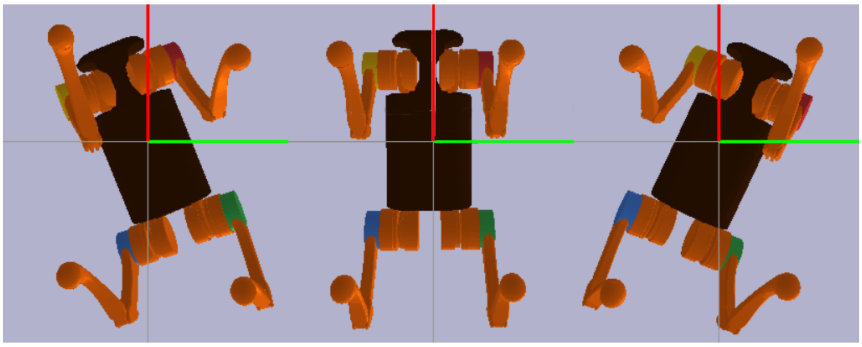


Figure 3.3: Yawing

For the sake of observing the locomotion of the quadruped robot more intuitively, we chose the bottom view as the observation view in the screenshots of yawing. In the initial position, the body of the quadruped robot was parallel to the longitudinal axis, which means that the yaw angle was 0. While the slider of debug parameter was pulled to the left, the yaw angle became negative, and the robot's body made counterclockwise rotation about the normal axis. While the slider of debug parameter was pulled to the right, the yaw angle became positive, and the robot's body made clockwise rotation about the normal axis.

It can be seen that the leg state of the robot is diagonally symmetrical, that is, the joints' state of the front-left leg is similar to that of the hind-right leg, and the joints' state of the front-right leg is similar to that of the hind-left leg. While the robot yawing to right, the front legs rotated counterclockwise, and the hind legs rotated clockwise, from the front perspective, relative to the abduction/adduction hip joints.The front-right leg and hind-left leg were crooked. The front-left leg and hind-right leg were extended. The state of these joints is actually similar to the motion of leg joints in squatting.

3.4 Rolling

Rolling, for the quadruped robot, is the action of rotating about the longitudinal axis [?]. By pulling the custom slider of squatting to adjust the parameters, a series of actions of the quadruped robot when rolling are shown in Fig. 3.4.

For the sake of observing the locomotion of the quadruped robot more intuitively, we chose the front view as the observation view in the screenshots of rolling. In the initial position, the heights of the left and right hip joints from the ground

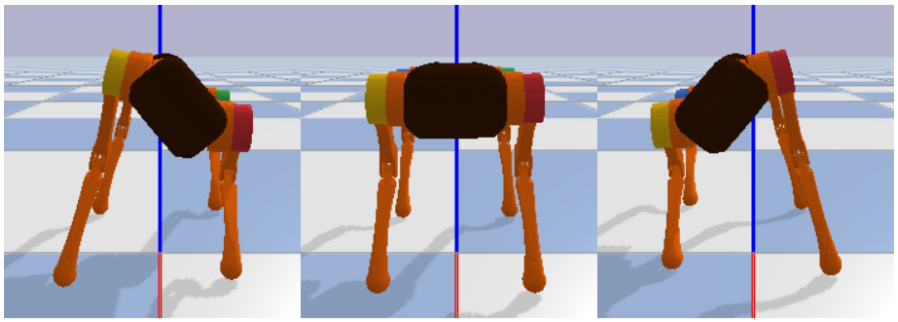
e 

Figure 3.4: Rolling

are the same, which means that the roll angle was 0. While the slider of debug parameter was pulled to the left, the roll angle became negative, and the robot's body made clockwise rotation about the longitudinal axis. While the slider of debug parameter was pulled to the right, the roll angle became positive, and the robot's body made counterclockwise rotation about the longitudinal axis.

The change in the joints' state of rolling is similar to pitching, except that the leg state of the robot is left and right symmetrical. While the robot pitched clockwise, the front legs rotated counterclockwise, and the hind legs rotated clockwise, from the front perspective, relative to the abduction/adduction hip joints. The right-side legs would be extended and the left-side legs would be crooked.

Chapter 4

Discussion and Conclusions

4.1 Limitations and Improvement

4.1.1 Ranges of Debug Parameters

In the simulation, the debug parameters were utilized to adjust the locomotion of the quadruped robot. The ranges of debug parameters are given in advance. The fixed ranges would not affect the robot when it performs a single action. Nevertheless, it is not appropriate for the robot to use the fixed range when performing multiple actions at the same time. For example, if the robot stretches its legs to reach the highest squatting position, which means the legs have reached the longest. Afterwards, moving other sliders will lead to program errors, and the robot will disappear in the GUI [**?**] owing to the errors. To avoid the error caused by the fixed range, the approach is to calculate the real-time debug parameter range according to the joint state of the robot. However, the real-time value ranges need to get the joints' state firstly, then bring them into the correlation matrix to deduce the value range inversely, and compare it with the value range of the joint itself. The complexity of real-time range is close to the quadrupedal locomotion, so it is hard for us to spare time to optimize this part of the code. Otherwise, there is another way to improve the robustness of the code, which is to narrow the range make it so that the robot cannot reach the limit range. Although the problem is not solved from the root, there is less possibility that the code will make mistakes during operation.

4.1.2 Non-zero steady-state error

When the robot is stationary, its joints are not completely static. The robot's joints would move in a minute range. The range of this steady-state error is approximately five decimal places. However, the current state read is used as a reference when controlling the attitude of the robot. Therefore, the previous

steady-state error will be substituted into the next action, and the program does not know the generation of this error. In the process of research, we find that as long as the accuracy of the parameters is controlled in three places after the decimal point, the steady-state error can be effectively reduced. But in theory, the problem of steady-state error should not be solved in this way. Therefore, in the following aspects that can be improved, we hope to reduce the steady-state error by using the robust integral controller [**?**] for the nonlinear system.

4.1.3 Friction between the plane and robot

When dragging the slider rapidlly to adjust the attitude of the robot, the robot will shift from the original position. This is because the friction between the toe joints of the robot and the plane is too small to provide the necessary force for the robot to move under this acceleration. The solution is to increase the friction coefficient between the toe joints and the plane in order to increase the friction force.

4.2 Future work

4.3 Intellectual property

4.4 Completion degree of objectives

4.5 Conclusion

Appendices