Balance control of quadruped robot based on model predictive control

ChengYe Wu 1*, Qing Wei 1, Cong Zhang 1 and HongLei An 1

1 Department of Intelligent Robotics, National University of Defense Technology, Changsha Hunan Province, 410000, China.

*wuchengye@nudt.edu.cn

Abstract. Aiming at the steady motion of the quadruped robot during support period of the locomotion, it is necessary to plan the ground reaction force (GRF) of legs. For the control of the plane height of the robot body, we take the standing balance of the simply modelled quadruped robot as the research object, and use the model predictive control algorithm to plan quadruped robot’s GRF of its position and attitude control, and then map the force to the operating space through the Jacobian matrix of the joint space to obtain the torque of the joint space. This paper uses MATLAB and RecurDyn for the co-simulation, and obtains good control results from it. At the end, we analyze the feasibility and rationality of its algorithm application by comparing it with the traditional balance controller. The work above provides the basis for the control and design of the subsequent quadruped robot motion planning.

1. Introduction

As we know, mobile robots have good application prospects and extensive social needs in the field of energy exploration, disaster relief, and military. Compared with wheeled robots and crawler robots, quadruped robots have unparalleled advantages and have good sports performance in complex terrain environment. During the movement, a good balanced state is a very important evaluation index for the performance of the robot. Therefore, a corresponding balance controller must be designed during the locomotion of the robot. The control of GRF between the robot’s legs and the environment is particularly important to the smooth control of the robot. Controlling the robot’s legs can reduce the energy consumption of the legs and the mechanism load, and can also adapt to changes in the ground environment to better achieve the movement step. In this regard, both home and abroad have carried out in-depth research and achieved a lot of research results. Briggs and Klein used active compliant control methods to control the foot force of the robot [1]; Nagy et al. proposed a walking control method for foot-to-environment contact [2], which is composed of two algorithms: appropriate changes in the environment and foot force distribution; Gorinevsky proposed the foot of the walking robot on hard ground and elastic pavement force control method [3]; for the harsh environment, Yoneda et al. proposed a foot model force feedforward control and feedback control method based on the suspension model [4] to stabilize the expected posture of the body; for inhibiting the disturbance when the foot of the leg touches the ground, Huang and Fukuhara proposed a sliding mode control method based on virtual suspension model [5]; For the force control of the foot robot, StarlETH [6] was applied with the traditional method based on the body attitude and joint angle, and achieved good results, it could perform walk, trot and running trot, and can smooth transition between those gaits.
The HYQ [7] quadruped robot achieved a smooth control of the quadruped robot by establishing a virtual model and using PD control of the wrench. MIT cheetah3[8,9] and minicheetah[10] proposed a method to obtain ground reaction force based on model predictive control, and achieved good results. The robot could perform various gaits under the framework of model predictive control. This paper takes the bionic quadruped robot as the research object, and uses model predictive control to predict the force at the foot end, and then maps it to the joint space via the Jacobian matrix in the operation space to obtain the joint torque. In this paper, by using MATLAB and RecurDyn simulation software for co-simulation, the analysis of the quadruped robot balance control provides important support for the subsequent research work. The structure of this paper is as follows: The second section is mainly for dynamic modelling of the quadruped robot; The third section is the design of controllers; The fourth section is the simulation analysis of quadruped robots; The fifth section is the conclusion.

2. Dynamic modelling of quadruped robot

The four-legged robot in the simulation is shown in the Figure 1. The robot is composed of a body and four legs, each leg is connected by three links, and has three degrees of freedom, therefore the entire robot has 12 degrees of freedom. The mass of the leg is negligible relative to the mass of the body. The Figure 1 shows the definitions of the model and related parameters.

Figure 1. Definitions of physical model and related parameters.

(whereas \( r_i \) is displacement of body centroid to foot position, \( f_i \) is Foot contact ground reaction force, \( \dot{x}_{xyz} \) is the Inertial coordinates, \( \dot{a}_{xyz} \) is the Body coordinate )

According to the floating base system, the following dynamic model can be established, with body attitude angle, centroid position, body angular velocity, and body centroid acceleration as the state variables. Assuming that the pitch angle and roll angle of the system are approximately zero in the simulation, the system dynamics can be modelled as follows. Where \( \Theta = [\phi, \theta, \psi] \), \( \psi \) is the yaw angle, \( \theta \) is the pitch angle, \( \phi \) is the roll angle.

\[
\begin{bmatrix}
\dot{x} \\
p \\
\omega \\
\dot{\omega}
\end{bmatrix} = \begin{bmatrix}
\Theta \\
\end{bmatrix}
\]

(1)

From the overall acceleration of the system, we can get:

\[
\ddot{p} = \left( \sum_{i=1}^{n} f_i \right) / m - g
\]

(2)

The rotation torque of the robot can be obtained:
\[
\frac{d}{dt}(I\omega) = \sum_{i=1}^{n} r_i \times f_i = I\dot{\omega} + \omega \times (I\dot{\omega})
\]  
(3)

In formula (3), \(I\) is the body's inertial tensor.

\[
I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}
\]  
(4)

The inertial tensor of the body relative to the inertial tensor of the inertial system can be obtained by the coordinate rotation: \(\hat{I} = R_x(\psi)R_y(\theta)R_z(\phi)\), whereas \(\hat{I}\) is the tensor in the inertial frame. The relationship between the system attitude angle, the system body coordinate system and inertial coordinate system can be obtained:

\[
\hat{R} = [\omega] \times R
\]  
(5)

\[
\omega = \begin{bmatrix} \cos(\theta) \cos(\psi) & -\sin(\theta) \cos(\psi) & \sin(\psi) \\ \cos(\theta) \sin(\psi) & \sin(\theta) \sin(\psi) & \cos(\psi) \\ 0 & 0 & 1 \end{bmatrix}
\]  
(6)

\[
R = R_x(\psi)R_y(\theta)R_z(\phi)
\]  
(7)

In the system simulation, the pitch angle and roll angle can be approximately zero, then the above formulas can be approximated:

\[
\frac{d}{dt}(I\omega) = I\dot{\omega} + \omega \times (I\dot{\omega}) \approx I\dot{\omega}
\]  
(8)

\[
\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = R_z(\psi)\omega = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \omega \end{bmatrix}
\]  
(9)

\[
\hat{I} = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]  
(10)

From the dynamic analysis above, the system dynamics state space equation of the system can be obtained:

\[
\frac{d}{dt} \begin{bmatrix} \Theta \\ p \end{bmatrix} = \begin{bmatrix} 0_3 & 0_3 & R_z(\psi) & 0_3 \\ 0_3 & 0_3 & 0_3 & 1_\rho \\ 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \begin{bmatrix} \Theta \\ p \end{bmatrix} + \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} + \begin{bmatrix} g \end{bmatrix}
\]  
(11)

Rewrite \(\begin{bmatrix} 0 & 0 & 0 & g \end{bmatrix}^T\) this term as a state quantity \(G\) of the system, so that the dynamic state space equation can be written as:

\[
\frac{d}{dt} \begin{bmatrix} \Theta \\ p \\ \omega \\ \rho \\ G \end{bmatrix} = \begin{bmatrix} 0_3 & 0_3 & R_z(\psi) & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 1_\rho \\ 0_3 & 0_3 & 0_3 & 0_3 & 1_\rho \\ 0_3 & 0_3 & 0_3 & 0_3 & 1_\rho \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \begin{bmatrix} \Theta \\ p \\ \omega \\ \rho \\ G \end{bmatrix} + \begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix}
\]  
(12)
3. Controller design

In this paper, model predictive control is used. Only the simple low-speed motion is considered, and the influence caused by the sudden change of the control quantity can be ignored. At the same time, the optimized control quantity can be directly applied to the control system to predict the next state quantity. The control flow of this method is composed of discretization of prediction model, prediction iteration, and optimization solution.

3.1. Linear discretization of prediction model

The linear discrete state space equation of the system is

\[ x(k+1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} T^* R_1(\psi) \\ T^* R_2(\psi) \\ T^* R_3(\psi) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} u(k) \]

\[ x(k+1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} u(k) \]

(13)

3.2. Forecast iteration

Assuming that the system has \( N \) discrete steps from time \( t_0 \), it can be iteratively obtained according to the above discrete state equation \( (13) \):

\[ x(k+N) = \sum_{i=0}^{N-1} \begin{bmatrix} A^{N-i} \\ A^{N-i} \\ \vdots \\ A^{N-i} \end{bmatrix} \begin{bmatrix} B \\ A^2 B \\ \vdots \\ A^{N-1} B \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N-1) \end{bmatrix} \]

(14)

Then the related state variables and control variables are combined and written in the form of state space equations, so there is the following state equations:

\[ \begin{bmatrix} x(k+1) \\ x(k+2) \\ x(k+3) \\ \vdots \\ x(k+N) \end{bmatrix} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix} \begin{bmatrix} x(k) \\ AB \\ A^2 B \\ \vdots \\ A^{N-1} B \end{bmatrix} + \begin{bmatrix} B \\ AB \\ A^2 B \\ \vdots \\ A^{N-1} B \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k+N-1) \end{bmatrix} \]

(15)

Then the state space equation of the above iteration can be written as:

\[ X = C + \Psi U \]

3.3. Quadratic optimization

Starting from a certain moment \( t_0 \), assuming that the value of the reference state variable at the moment \( t_0 \) is \( \eta(k) \), the state at the Nth discrete time is \( \eta(k+N) \).

Get:
It can be seen from the above matrix, where $U$ is the optimized variable matrix, and $C\Psi$ are the constant matrix. The following takes the difference matrix $E$ of the state variables of the simulation time and state variables of the reference time, so the difference matrix $E$ is as follows:

$$E = X - \gamma = \begin{bmatrix}
    x(k+1) - \eta(k+1) \\
    x(k+2) - \eta(k+2) \\
    x(k+3) - \eta(k+3) \\
    \vdots \\
    x(k+N) - \eta(k+N)
\end{bmatrix}$$

(16)

Taking the objective optimization function as $J = E^TQE + U^TRU$, we can obtain:

$$E = C - \gamma + \Psi U$$

$$E^T = (C - \gamma)^T + U^T\Psi^T$$

(18)

Bringing the above equations (18) into the objective optimization function, we can get:

$$J = \left((C - \gamma)^T + U^T\Psi^T\right)Q(C - \gamma + \Psi U) + U^T RU$$

(19)

The equation (19) can be as follows:

$$J = (C - \gamma)^T Q(C - \gamma) + U^T\Psi^T Q(C - \gamma) + (C - \gamma)^T Q(\Psi U) + U^T\Psi^T Q\Psi U + U^T RU$$

(20)

In the above formula (20), it can be seen that the first term is constant and can’t be considered in the objective optimization function. The second term and the third term are transposed to each other, and are numerically equal. The fourth and fifth terms are related with the optimization control variable. We can integrate terms which contain control variables into new equations:

$$J^* = 2(C - \gamma)^T Q(\Psi U) + U^T (\Psi^T Q\Psi + R) U$$

(21)

It can be seen from the equation (21) that obtained target optimization function is convex quadratic, and the quadprop function in MATLAB can be used to solve the optimal control variables of the above target function (21). The standard quadprop function in MATLAB is as follows:

$$\min f(x) = \frac{1}{2} U^T H x + c U$$

$$H = 2(\Psi^T Q\Psi + R)$$

$$c = 2(C - \gamma)^T Q\Psi$$

(22)

Convert the state space equation of the quadruped robot to a quadratic optimization function:

$$J(x, f) = \min \sum_{i=0}^{k-1} (x_{r+1} - x_{r+1, ref})^T Q(x_{r+1} - x_{r+1, ref}) + f_i^T R f_i$$

(23)

(Whereas $f_i$ is the control variable, In the model, and matrix $Q$ and $R$ are the weight matrix of state variable and control variable respectively)
By replacing the form \( x_{i+1} \) with \( x_i \), the cost function can be rewritten as:

\[
J(x, f) = (A_{\xi_i} + B_f - x_{i+1,ref})^T Q(A_{\xi_i} + B_f - x_{i+1,ref}) + f_i^T R f_i
\]

\[
= (A_{\xi_i} + B_f)^T Q(A_{\xi_i} + B_f) - 2(A_{\xi_i} + B_f)^T Q x_{i+1,ref} + x_{i+1,ref}^T Q x_{i+1,ref} + f_i^T R f_i
\]

\[
= \frac{1}{2} f_i^T (2B^T Q B_f + R) f_i + 2(A_{\xi_i})^T Q B_f f_i + (A_{\xi_i})^T Q(A_{\xi_i})
\]

\[
= -2(A_{\xi_i} + B_f)^T Q x_{i+1,ref} + x_{i+1,ref}^T Q x_{i+1,ref} + f_i^T R f_i
\]

\[
= \frac{1}{2} f_i^T (2B^T Q B_f + R) f_i - 2B^T Q(A_{\xi_i} - x_{i+1,ref}) f_i + M
\]

whereas \( M \) is the irrelevant items of control variables \( f_i \).

\[
M = (A_{\xi_i})^T Q(A_{\xi_i}) + 2(A_{\xi_i})^T Q x_{i+1,ref} + x_{i+1,ref}^T Q x_{i+1,ref}
\]

Then according to the standard form of quadratic optimization, we can get:

\[
\begin{cases}
H_i = 2B^T Q B_f + R \\
C_i = 2B^T Q(A_{\xi_i} - x_{i+1,ref})
\end{cases}
\]

4. Co-simulation and analysis

For the established model, we use MATLAB and RecurDyn for co-simulation. The overall simulation structure diagram as follows:

Figure 2. MATLAB and RecurDyn co-simulation diagram. (MATLAB version is 2019a, RecurDyn version is V9R1)

In the simulation experiment, this article sets the balance position of the quadruped robot above the origin of the inertial coordinates, the main parameters are shown in the following table:

| Table 1. Parameters for the simulation environment |
|-----------------------------------------------|
| **Unit** | **Value** |
| Equilibrium position | m | (0,0,0.277) |
Body inertia tensor \( \text{kg} \cdot \text{m}^2 \) Ixx=0.194; Iyy=0.697; Izz=0.837

Body quality kg 50.24

Leg length m L1=0.04m; L2 = 0.128m; L3 = 0.128m

Leg quality kg m1=0.0986kg; m2=0.316kg; m3=0.3487kg

friction factor \( \mu \) 0.8

The following figures show the results of the quadruped robot using model predictive control:

Figure 3. Actual displacement and expected displacement of body centroid in MPC control.

Figure 4. The output of Leg1 Z direction expected force and actual force in MPC control.

Figure 5. The output of Leg2 Z direction expected force and actual force in MPC control.

Figure 6. The output of Leg3 Z direction expected force and actual force in MPC control.

Figure 7. The output of Leg4 Z direction expected force and actual force in MPC control.
The following figures show the results of the quadruped robot using PD balance control[7]:

Figure 8. Actual displacement and expected displacement of body centroid in PD control.

Figure 9. The output of Leg1 Z direction expected force and actual force in PD control.

Figure 10. The output of Leg2 Z direction expected force and actual force in PD control.

Figure 11. The output of Leg3 Z direction expected force and actual force in PD control.

Figure 12. The output of Leg4 Z direction expected force and actual force in PD control.

From the above position control diagram, especially the position control in the Z direction, the results of the traditional PD balance controller has large fluctuations, the maximum fluctuation is 6%, and the model predictive control basically meets the expectations, and the relative fluctuations is smaller, it can be seen that the effect of the model predictive control is more stable than that of the traditional controller; and from the force output, the foot force output by the model predictive control is closer to the expected force given by the algorithm, and the up and down fluctuations are relatively small. The effect of PD balance control is slightly worse. In summary, the performance of the model predictive controller is better than traditional controllers.

5. Conclusion
In this paper, the model predictive control algorithm is used to simulate the four-legged support of the quadruped robot. At the same time, it is compared with the traditional PD balance controller. By
comparison, the model prediction is superior to the traditional controller in control performance. Therefore, the model predictive control method analyzed in this paper plays a very important role in the subsequent movement control of the quadruped robot, and will be applied in future research.

References

[1] Klein C A, Briggs R L. Use of Active Compliance in the Control of Legged Vehicles[J]. Systems Man & Cybernetics IEEE Transactions on, 1980, 10(7): 393-400.

[2] Nagy P V, Whittaker W L, Desa S. A walking prescription for statically-stable walkers based on walker/terrain interaction[C]. IEEE Int. on Robotics & Automation, 1992.

[3] Gorinevsky D M, Shneider A Y. Force Control in Locomotion of Legged Vehicles over Rigid and Soft Surfaces[C]. Int. on Advanced Robotics, 1990.

[4] Kan Y, Iiyama H, Hirose S. Sky-Hook Suspension Control of a Quadruped Walking Vehicle[C]. IEEE Int. on Robotics & Automation, 1994.

[5] Huang Q. Posture and Vibration Control Based on Virtual Suspension Model for Multi-Legged Walking Robot[C]. IEEE/RSJ Int. on Intelligent Robots & Systems, 2007.

[6] Gehring C, Coros S, Hutter M, et al. Control of Dynamic Gaits for a Quadrupedal Robot[C]. IEEE International Conference on Robotics and Automation (ICRA) IEEE Int. on Robotics and Automation (ICRA), 2013.

[7] Winkler A W, Mastalli C, Havoutis I, et al. Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain[C]. 2015 IEEE Int. on Robotics and Automation (ICRA), 2015.

[8] Bledt G, Powell M J, Katz B, et al. MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot[C]. IEEE Int. of Intelligent Robots (IROS) 2018, 2018.

[9] Carlo J D, Wensing P M, Katz B, et al. Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control[C]. 2018 IEEE/RSJ Int. on Intelligent Robots and Systems (IROS), 2018.

[10] Kim D, Di Carlo J, Katz B, et al. Highly Dynamic Quadruped Locomotion via Whole-Body Impulse Control and Model Predictive Control[J], 2019.

[11] Camacho E F. Book reviews: Predictive control with constraints[J].