Omnidirectional Motion Control for the Humanoid Soccer Robot

Zhechao Gao, Jia Wu, Hongwu Chen, Bowen Pan,
Hao Zheng, Dingwen Liang, and Xiaoming Liu

Faculty of Computer Science, China University of Geosciences,
Wuhan 430074, P.R. China
wujiawb@126.com

Abstract. In RoboCup 2008 World Championship, the model of RoboCup 3D simulation has changed from Soccerbot to Nao. A series of parameters have been changed in a Nao model. This has greatly increased the complexity and difficulty of motion controller for Nao. Based on the analysis of Nao's structure, our team G-Star has worked out the quantitative relation of joint angle in motion control and developed a toolkit to calculate the angle of joints accurately. In the experiment on RoboCup 3D server platform, our robots can stand up promptly and walk smoothly, which is fast enough to meet the real-time requirements. In this paper, we will give a detailed description of the architecture in basic motion using Nao model.

Keywords: RoboCup 3D, Humanoid robot, Motion control, Simulation.

1 Introduction

In humanoid robot simulation like RoboCup 3D, the controller of robot motion is prior to the decision system. To make a strategy more flexible and stable, a set of basic actions such as walking, turning, standing up is indispensable. Compared to Soccerbot, Nao, although they are both humanoid robot, has different structure in some joints, which adds joint angle limit and changes some parameters of joints.

Through the analysis of Nao's skeleton structure, we have found a simple principle of basic action under Nao model, which makes our robots walk faster [1] on the field and stand up promptly. In our motion design, the robot can also walk backward [2] as same as walk forward which avoids the time loss of turning and makes the strategy more effective [3].

In this paper, we describe a particular analysis of Nao model and propose a basic motion control approach. Then we give a snapshot of our development toolkit called RoboCup3D-Tool which is helpful to quantitative computing under the motion control of Nao model.

The rest of the paper is organized as follows. In Section 2, we give a formal description of the structure of Nao model and give a preliminary analysis of joint control. In Section 3, we mainly discuss our motion control mechanism. A snapshot of our development tool is presented in Section 4. Section 5 concludes the whole paper and gives some future works.

Z. Cai et al. (Eds.): ISICA 2009, CCIS 51, pp. 1–8, 2009.

2 Preliminaries

2.1 Nao Model

As we can see the Nao's skeleton from Fig. 1, there are five joints [4] can influence the action of leg on one side. Three of them are hinge joints which are most important in controlling forward and backward direction. In the figure, they are joint leg3 which join the robot's body and thigh, joint leg4 which join the robot's thigh and shank and joint leg5 which join the robot's shank and foot. The leg can turn to any configuration with the three joints. What's more, we can achieve more actions in landscape orientation if we use joint leg2 and leg6 which control the offset of transverse.

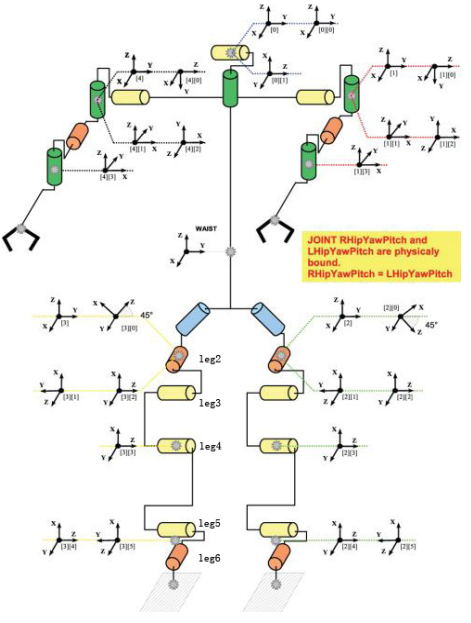

Fig. 1. The joints of Nao robot

2.2 Relations between Joints

According to the joints structure, we choose a suitable initialization angle for the five joints, so that all the other actions can start with this state.

Here, to make the analysis easier to understand, we simply transfer the three-dimensional [5] reference frame into two-dimensional reference frame. We can see the two-dimensional plane in Fig. 2 and Fig. 3, which show the quantitative relationship of the leg and the whole body.

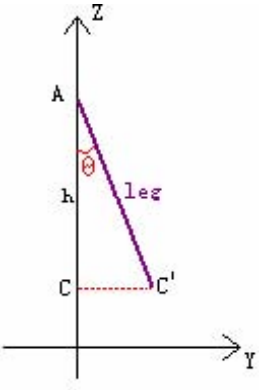

Fig. 2. Quantitative relationship of robot leg

In Fig. 2, we can get the transverse relations of angles and lengths as follows:

$$h = AC' \times \cos\theta. \tag{1}$$

$$dy = CC' = AC' \times \sin\theta. \tag{2}$$

Where h represents the height of the robot's lower body, and dy represents the leg's translation in y direction.

In Fig. 3, we can get the joint angle relations as follows:

$$h = l_{thigh} \times \sin c + l_{shank} \times \sin d. \tag{3}$$

$$dx = l_{thigh} \times \cos c - l_{shank} \times \cos d. \tag{4}$$

$$\angle a + \angle leg3 = 180^{\circ}. \tag{5}$$

$$\angle c + (-\angle leg4) = 180^{\circ}. \tag{6}$$

$$\angle e + \angle leg5 = 90^{\circ}. \tag{7}$$

$$\angle body\_ang = \angle b. \tag{8}$$

$$\angle b + \angle a = 90^{\circ} + \angle c. \tag{9}$$

$$\angle d = \angle e. \tag{10}$$

Where dx represents the leg's translation in y direction, l_{thigh} and l_{shank} represents the length of the robot thigh and shank respectively, and the $body\_ang$ represents the gradient of the robot body.

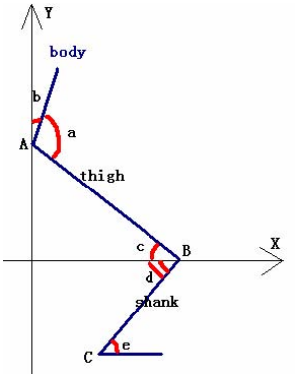

Fig. 3. Quantitative relationship of the whole body

According to the ten equations above, we can figure out the angles of leg2, leg3, leg4, leg5 and leg6 with dx, dy, h. So we can write a function which maps from the inputs dx, dy and h to the outputs all the degrees of the joints above. This is a fundamental function of our motion control system, which keeps the robot body's balance and COM (Center of Mass) smooth. Then, we can use it as a basic control in many action sets.

3 Motion Control

3.1 Moving

Fig. 4 shows the two states in the moving action, the left leg and the right leg. There are two steps in each leg, which hold the legs up and down, stride and stop. The step 1 represents the stride action, while the step 2 is the dropping action. With the move function, we can control actions with the inputs dx, dy, h of each leg.

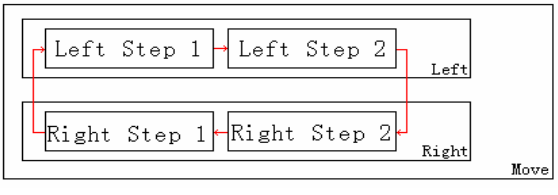

Fig. 4. The process of moving

3.2 Walking

If both of the two legs' dy are 0 and dx are the same, the robot will walk smoothly. Where the positive value leads to forward walk and negative to the backward.

3.3 Turning

3.3.1 Turning When Walking

It's easy to do this job if you add turning while walking straightly. The robot will walk in a roundness path [6]. The relation of step lengths and radius is as follows:

$$2\pi r_1 / 2\pi r_2 = dx_l / dx_r. \tag{11}$$

$$r_2 - r_1 = x. \tag{12}$$

$$r = (r_1 + r_2)/2 = ((dx_l + dx_r) \times x) / (|dx_l - dx_r| \times 2). \tag{13}$$

Where x is the distance between the center of feet, dx_l, dx_r represent the displacement of left and right leg in x direction. Fig. 5 shows the relation which is described above.

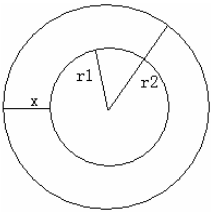

Fig. 5. The relation of turning and walking

3.3.2 Turning without Walking

We can easily get the function from the equations above. That is, if r is zero, the robot stays. So when dx_l plus dx_r equals 0, we will get action of turning without walking.

3.4 Transverse Move

Just if the dy isn't zero, the robot will do transverse moving. You will get more wizardly actions [7] if add other actions.

3.5 Symmetrical Robot Actions

Not a new action but an idea of design. Our robot can do everything with either its face or back. So it will be smarter and faster than ever before. In this way robot saves more time when adjusting [8], which makes our robot easier to get the ball. Also it has advantages in dynamic environments, which has been proved in RoboCup 2008 China Open. Fig. 6 gives a snapshot when using this method.

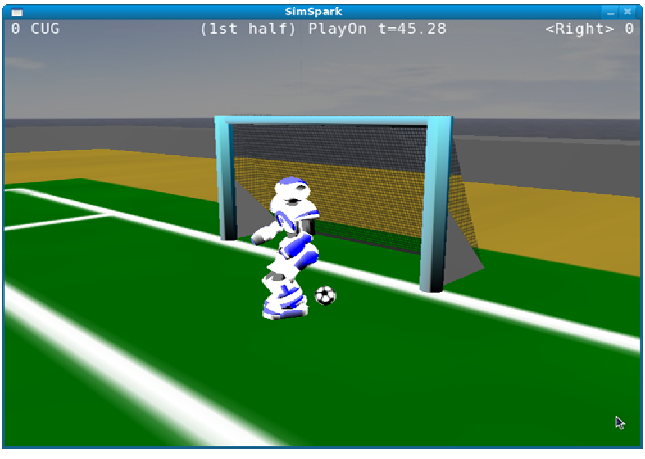

Fig. 6. Walk backward for goal

4 Development Tool

RoboCup3D-Tool is written in C# which consists of four modules: initial module, walk module, transverse move module, turning module. Hera, we will take walk module for presentation.

4.1 Initial Module

Fig. 7 is a snapshot of RoboCup3D-Tool. The picture on the left-top is an initial state of the robot. The other modules are based on the initial state.

The initial state angle is: Leg3 = 35.50, Leg4 = -63.90, Leg5 = 28.40.

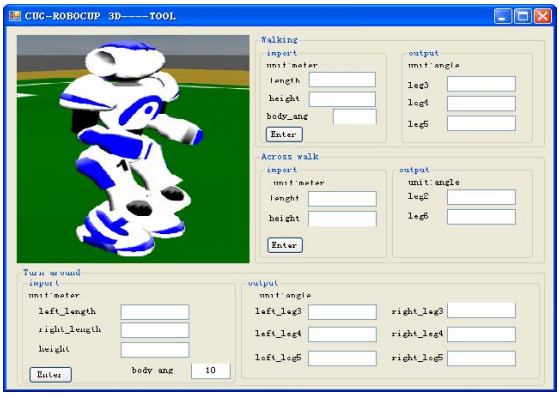

Fig. 7. The main window of RoboCup3D-Tool

4.2 Walk Module

Walk module is the core of the whole action which consists of two sub-window import and output, which can be seen in Fig. 8.

Import:

Length: the distance between foot and torso (length > 0, forward; length < 0, backward).

Height: the distance between foot and field.

Body_ang: the slant angle of torso.

Output:

Leg3, Leg4, Leg5

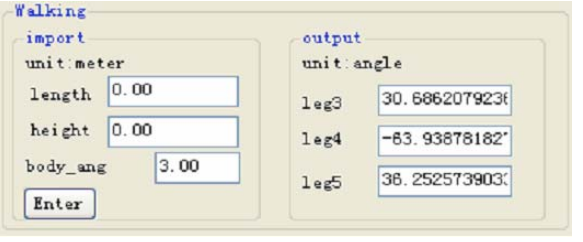

Fig. 8. Walk module

5 Conclusion and Future Works

In experiment, our robots are able to fast walk, and the speed of standing up is also prompt. The principle of walking forward or backward is totally symmetric, so that while robot's direction is against the goal, it can also carry the ball backward quickly. In the future, we will improve our adjusting efficiency between walking and stopping to reduce the time loss in motion transforming.

Acknowledgments

We very thank Mr. Xuesong Yan for his kindly guidance and help.

References

1. Collins, S.H., Wisse, M., Ruina, A.: A 3D Passive — Dynamic Walking Robot with Two Legs and Knees. The International Journal of Robotics Research 20(7), 607–615 (2001)
2. Behnke, S.: Online Trajectory Generation for Omnidirectional BipedWalk. In: Proceeding of the 2006 IEEE International Conference on Robotics and Automation, Orlando, Florida (2006)
3. KÖgler, M.: Simulation and Visualization of Agents in 3D Environments. Diploma Thesis, 13–25 (2003)
4. Kajita, S., Hirukawa, H., Yokoi, K., Harada, K.: Humanoid Robots (Chinese Edition). Tsinghua University Press, Beijing (2007)

5. Lattner, A.D., Rachuy, C., Stahlbock, A., Visser, U., Warden, T.: Virtual Wender 3D team documentation. Technical Report 36, TZI - Center for Computing Technologies, Bremen (2006)
6. Niehaus, C.: Optimierung des omnidirektionalen Laufens eines humanoiden Roboters. Master's thesis, Bremen (2007)
7. Röfer, T., Budelmann, C., Fritsche, M., Laue, T., Müller, J., Niehaus, C., Penquitt, F.: B-Human team description for RoboCup 2007 (2007)
8. Wagner, T., Bogon, T., Elfers, C.: Incremental generation of adductive explanations for tactical behavior. In: Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.) RoboCup 2007. LNCS (LNAI), vol. 5001, pp. 401–408. Springer, Heidelberg (2008)