Development of a Humanoid Robot Simulator and Walking Motion Analysis

Noboru Sugiura[1], Masaki Takahashi[2]

[1] Dept. of System Design Engineering, Keio University
3-14-1, Hiyoshi, Kohoku-ku, Yokohama, 223-8522 Japan
sugiura.noboru@techno-road.com
[2] Dept. of System Design Engineering, Keio University
3-14-1, Hiyoshi, Kohoku-ku, Yokohama, 223-8522 Japan
takahashi@sd.keio.ac.jp

Abstract. We are developing a humanoid robot simulator based on ODE (Open Dynamics Engine). There exist a lot of dynamics simulation software packages now. However, we want to adjust the ODE's various physics parameters, therefore we have decided to develop a simulator using ODE's API and Visual C++ 2005. Now, we have built a humanoid robot simulator including several functions. In this paper, we report progress on the humanoid simulator and walking motion analysis.

Keywords: Humanoid Robot, Physics Simulation, ODE, Walking Motion, Linear Inverted Pendulum

1 Introduction

1.1 Humanoid Robot

Recently, various humanoid competitions are held in Japan. Especially, after ROBO-ONE [1,2] was held in Japan in the year 2002, a lot of amateur robot builders have developed humanoid robots. In cooperation with ROBO-ONE competitors, some of the robot companies have developed robot kits which are about 100,000 yen, as for example KONDO [3], VSTONE [4], etc.

Under these circumstances it becomes easy to develop humanoid robot hardware. But it is difficult to develop humanoid robot software because between the software and the hardware (dynamics, sensor, dynamics, etc) mutually affect.

As in RoboCup [5,6,7] each robot must decide their behavior by themselves, control software becomes more complicated. It is desired to develop robot software on a simulator.

1.2 Physics Simulation

Recently, many physics APIs have been released, as for example, ODE [8], PhysX [9], Havok [10], SpringHead [11], etc. They make it possible to develop a robot simulator at a low cost or free. Most of the physics APIs offer various physics functions, i.e. rigid body dynamics, joints between rigid bodies, collision, friction, etc. It is possible to develop a robot simulator with these physics API.

At the same time, many simulation application software packages have been released, for example, WEBOTS [12], UsarSim [13], Robotics Studio [14], Open HRP [15,16], SimPark [17], etc. They are based on the physics API or their original physics algorithm.

There are some papers about humanoid robot simulation using these simulators. HOAP-2 which is developed by Fujitsu was successfully simulated on WEBOTS [18]. Robovie-M which is developed by VSTONE was successfully simulated on UsarSim and Robotics Studio [19].

1.3 Development of a Simulator

There exist a lot of simulation packages now. However, to develop robot simulator, it is necessary to create detail model. For example, actuator model parameters are maximum torque, servo gain, back EMF, friction, etc. Sensor model parameters are band-width, digital resolution, sampling-delay, etc. ODE’s contact parameters are ERP, CFM, Mu, bounce, etc. To create and adjust these detail models, we have decided to develop an original simulator using Visual C++ 2005 and ODE's API.

On this paper, we report progress of development of humanoid robot simulator. We have developed humanoid robot model, robot motion editor, gyro sensor and inclination sensor model, and we have simulated walking motion.

2 Development of Humanoid Robot Simulator

2.1 Software Libraries

We are developing a humanoid robot simulator using ODE[20]. ODE is developed by Russell Smith. As ODE provides many physics functions, it is possible to develop a humanoid robot simulator. For 3-D computer graphics rendering, we use OpenGL. For a compiler, we use Visual C++ 2005 Express. All of these environments are available for free.

2.2 Modeling of Humanoid Robot

We designed a humanoid robot model as shown in Fig. 1. It has 20 degrees of freedom. All of the bodies are constructed by a simple box as Fig. 2(a), and all of the joints are hinge type. Total mass is 1.2 kg, height is 0.38 m. And there is a polygon view mode as shown in Fig. 2(b).

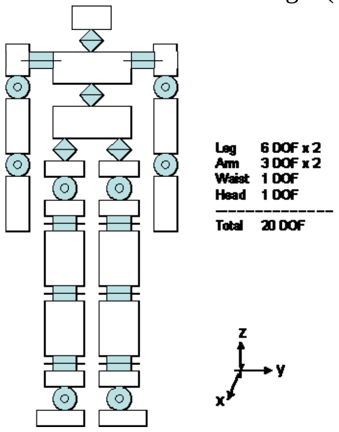
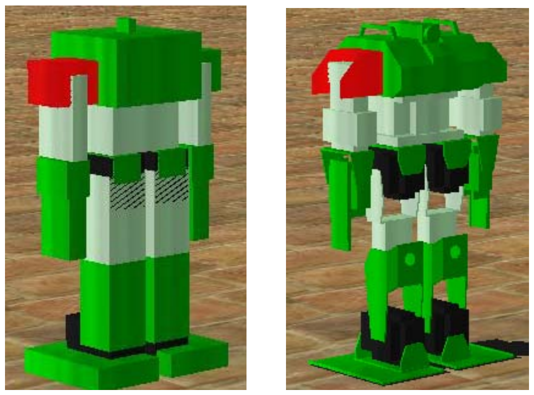


Fig. 1. Robot Model



(a) Box model (b) Polygon Model

Fig. 2. Modeling of Humanoid Robot

2.3 Robot Model

The robot model has many physical parameters (size, mass, etc.). In this simulator, it is possible to edit these parameters (Body position, Body rotation, Body mass, Joint position, Joint axis).

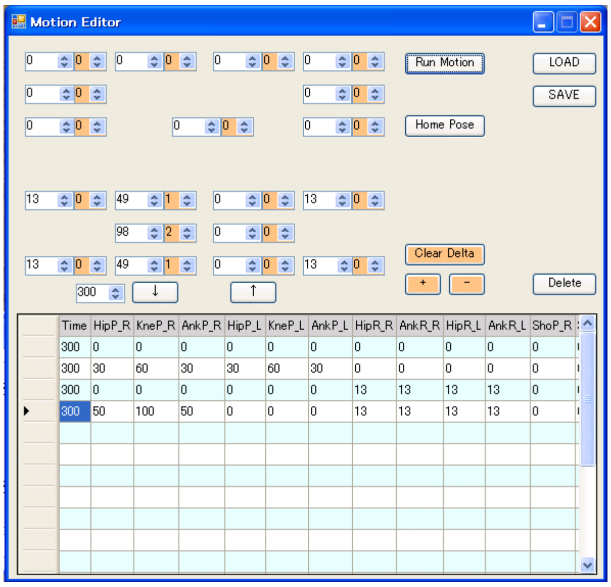
It is possible to import polygon data, too. This simulator supports STL format. STL format is one of the polygon data format. Most of the three-dimensional CAD system can export STL format. In this paper, we designed a robot by using Autodesk Inventor and exported the polygon data to STL files. Each STL file data is attached to each body. It is possible to edit STL data position, rotation and zoom.

2.4 Motion Editor

We design a motion editor as shown in Fig. 3. The motion editor has basic functions which most of the humanoid robot kits have.

The usage is as follows.

1) To make a pose, set each joint angle.
2) To make a motion, set the poses and transition time.
3) If the motion is executed, the joint angle values are linearly interpolated between poses.



Time	HipP_R	KneP_R	AnkP_R	HipP_L	KneP_L	AnkP_L	HipR_R	AnkR_R	HipR_L	AnkR_L	ShoP_R
300	0	0	0	0	0	0	0	0	0	0	0
300	30	60	30	30	60	30	0	0	0	0	0
300	0	0	0	0	0	0	13	13	13	13	0
300	50	100	50	0	0	0	13	13	13	13	0

Fig. 3. Motion Editor

2.5 Center of Gravity and Zero Momentum Point

In this simulator, it is possible to calculate CG (Center of Gravity) and ZMP (Zero Momentum Point). ZMP exists always in support polygon.

ZMP is calculated from torque and force at ankle joint [21,22]. In particular, when robot stands on right foot (Fig.4), ZMP ($\mathbf{p}_R = [P_{Rx} \quad P_{Ry}]$) is calculated as following.

$$p_{Rx} = (-\tau_y - f_x d)/f_x \qquad (1)$$
$$p_{Ry} = (\tau_x - f_y d)/f_y$$

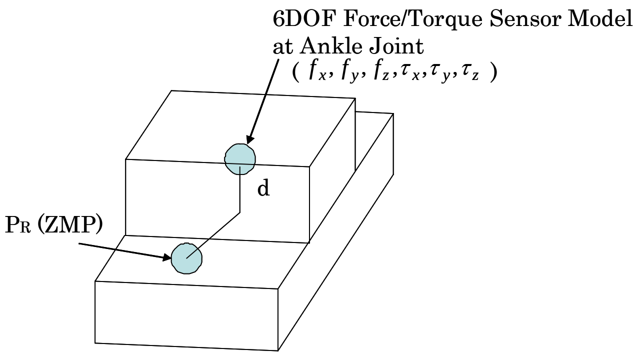

6DOF Force/Torque Sensor Model
at Ankle Joint
($f_x, f_y, f_z, \tau_x, \tau_y, \tau_z$)
d
PR (ZMP)

Fig. 4. ZMP

2.6 Gyro Sensor and Inclination Sensor

We implemented a gyro sensor and an inclination sensor model for this simulator, and to reduce some noise, both sensor models have a 2nd order low pass filter. Both of these sensors are important to control a robot.

A gyro sensor is used for a gyro feedback controller. The controller improves attitude stability. The controller efficiency is shown in section 3.3.

An inclination sensor is used for detecting attitude when a robot is fallen. After detecting falling, the behavior controller changes the control state to stand-up motion.

3. Walking Motion Analysis

3.1 Linear Inverted Pendulum

There exist various humanoid robot walking algorithms. Linear Inverted Pendulum [23] is one of the most well-known algorithms.

The basic Linear Inverted Pendulum method is as follows.

Considering the model as shown in Fig. 5 the equation of motion is

$$\begin{aligned} M\ddot{x} &= f\sin\theta \\ f &= Mg/\cos\theta \end{aligned} \tag{2}$$

and then

As g and z are constant value, x is described as following.

$$\begin{aligned} x(t) &= x(0)\cosh(t/T_c) + T_c\dot{x}(0)\sinh(t/T_c) \\ \dot{x}(t) &= x(0)/T_c\sinh(t/T_c) + \dot{x}(0)\cosh(t/T_c) \end{aligned} \tag{3}$$

$T_c = \sqrt{z/g}$. x direction(sagittal) and y direction(lateral) are described as same equation and each initial value is different.

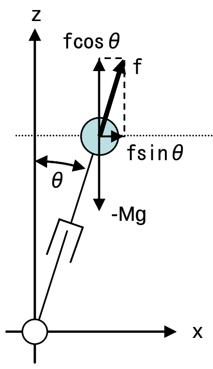
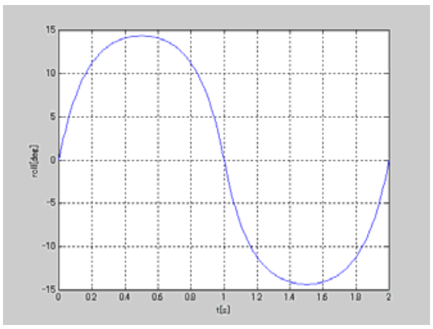


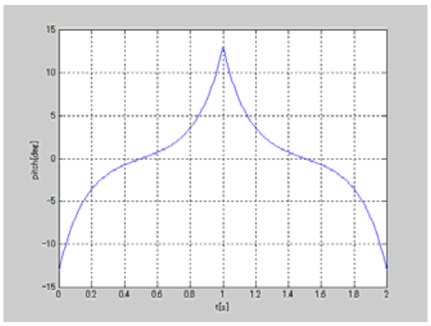
Fig. 5. Inverted Pendulum Model

3.2 Motion Pattern

Most of the robot kits are programmed based on motion patterns which are series of several pose data. This is a good interface to create various motions. We generated walking motion pattern from the theoretical Linear Inverted Pendulum algorithm. In particular, first, from Linear Inverted Pendulum algorithm, walking motion profiler is derived as Fig. 6(a). Second, we divide the profiler into eight points and approximate between points by linear interpolation as shown in Fig. 6(b). We have tried to divide it into four, but the walking performance is not stable. In general, a sine wave should be divided into about more than ten points.

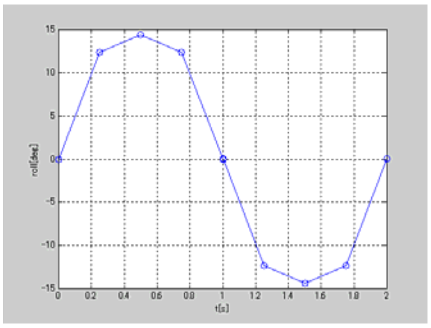


Roll Angle

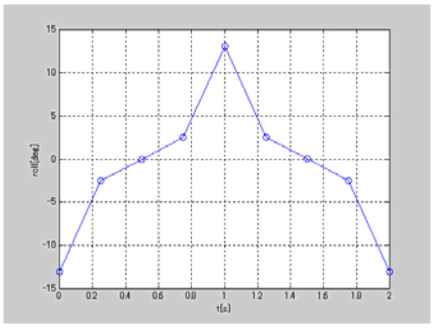


Pitch Angle

(a) Theoretical profile



Roll Angle



Pitch Angle

(b) Eight points profiler

Fig. 6. Walking Motion Profile

Workshop Proceedings of SIMPAR 2008
Intl. Conf. on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS
Venice(Italy) 2008 November,3-4
ISBN 978-88-95872-01-8
pp. 151-158

3.3 Gyro Feedback

Most of the humanoid robot kits have gyro sensors. They are used for gyro feedback controllers. The controller improves attitude stability. The control block diagram is shown in Fig. 7.

We simulated a walking pattern which is shown in Fig. 6. We compared this pattern both with and without gyro control. The result is shown in Fig.8. (a) is gyro control off, (b) is gyro control on. These figures show a trace of ZMP during walking. Comparing both figures, (b) shows stable results. This shows the effectiveness of gyro feedback control.

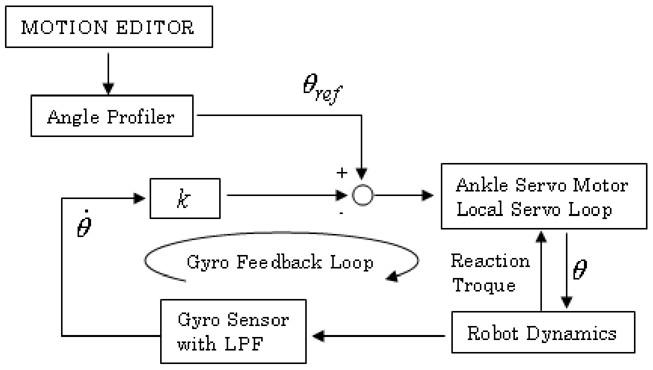
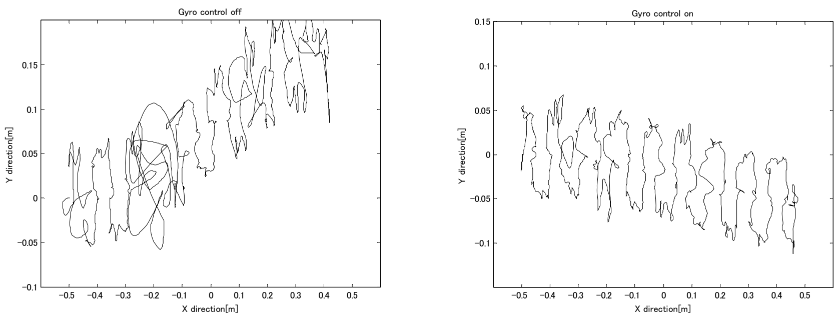


Fig. 7. Gyro Control Loop



(a) gyro control off (b) gyro control ofn

Fig. 8. ZMP trace during walking

4. Conclusion

In this paper, the progress of development of a humanoid simulator is reported. Our simulator is based on ODE and OpenGL, and compiled using Visual C++ 2005 Express.

The robot model parameters are editable and we built a motion editor to create various robot motions. This simulator calculates the CG and ZMP, and comes with an implementation of a gyro sensor and an inclination sensor. Both of these sensors are important to control a robot.

We simulated walking motion with motion pattern generated from Inverted Pendulum Model. We also evaluated the efficiency of using a gyro feedback controller.

This simulator is now under development and the latest version has more various functions. We want to go forward the development and want to report all functions as fast as possible.

References

1. ROBO-ONE. http://www.robo-one.com/
2. Mikio Azusa, 〞Biped Robot Kit Born by ROBO-ONE", Journal of the Robotics Society of Japan, Vol. 24 No. 1, pp.36～40, 2006
3. KONDO. http://www.kondo-robot.com/index.php
4. VSTONE. http://www.vstone.co.jp/
5. RoboCup http://www.robocup.org/
6. H. Kitano and M. Asada and Y. Kuniyoshi and I. Noda and E. Osawa and H. Matsubara, "RoboCup: A Challenge AI Problem", AI Magazine, 1997
7. H. Kitano and M. Asada, "The RoboCup humanoid challenge as the millennium challenge for advanced robotics", Advanced Robotics, 2000, vol.13, no.8, pp.723-736
8. ODE, http://www.ode.org/

9 PhysX, http://www.nvidia.com/object/nvidia_physx.html

10 Havok, http://www.havok.com/

11 SpringHead, http://springhead.info/index.en.php

12 WEBOTS, http://www.cyberbotics.com/

13 UsarSim, http://usarsim.sourceforge.net/

14 Robotics Studio, http://msdn.microsoft.com/ja-jp/robotics/default(en-us).aspx

15 Open HRP, http://www.is.aist.go.jp/humanoid/OpenHRP/jp/

16 Fumio Kanehiro, "OpenHRP: Open Architecture Humanoid Robotics Platform," International Journal of Robotics Research, vol.23, no.2, pp.155-165, 2004.

17 Oliver Obst and Markus Rollmann, “SPARK -- A Generic Simulator for Physical Multi-agent Simulations” Computer Systems Science and Engineering, vol.20, no 5, pp.347-356, 2005

18 Pascal Cominoli, "Development of a physical simulation of a real humanoid robot" Master’s thesis, EPFL, Swiss Federal Institute of Technology, 2005

19 Nicola Greggio, "3D models of Robovie-M in USAR Sim and Robotics Studio simulators", International Journal of Humanoid Robotics.

20 Russel Smith, "Open Dynamics Engine v0.5 User Guide", http://www.ode.org/ode-latest-userguide.html

21 M.Vukobratovic and J.Stepanenko: “On the Stability of Anthropomorphic Systems,” Mathematical Biosciences, Vol.15 pp.1-37, 1972.

22 Syuji Kajita, “Humanoid Robot”, Ohmsha, p.77, 2005

23 Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kazuhito Yokoi and Hirohisa Hirukawa, "The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation", Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, Hawaii, USA, Oct. 29 - Nov. 03, 2001