Team "ROPE"

[1] Ho Hoan Nghia, [2] Talasila Sateesh, Chew Chee Meng, Hong Geok Soon

Mechanical Engineering Department, National University of Singapore
[1]g0202612@nus.edu.sg, [2]g0203722@nus.edu.sg

Abstract. This paper presents some technical details about the hardware and software architecture of our humanoid robot ROPE. The control system of this robot has been built based on architecture of a PC system. This kind of system provides us with more power in programming than any other kind of microcontrollers. PC-104, which is an industrial PC with highly compact size, is used as central processor. The control program is developed in RT-Linux environment. This is a real-time operating system that allows us to control the robot in real-time with appropriate responsive behaviors.

1. Introduction.

Humanoid robot is becoming a very active research area with series of impressive introduction of commercial robots of several Japan companies like Honda, Sony, Fujitsu, etc…Those companies with strong financial and technological foundation are capable of producing high cost and custom-made components for their system. Even though we have less resource, we need to build a prototype to test our ideas with a hope that we can make useful contribution in this area. Our robot ROPE was designed out of that purpose.

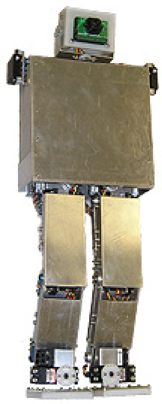

Fig. 1. ROPE-II

Most of the components used to build ROPE were bought off the shelf. The use of such components will help to reduce the cost of our robot. However, this fact does not restrict us from developing interesting behaviors for the robot. A proper design of architecture of control system will enable us to make a platform flexible enough for implementing certain behaviors for the robot and even accumulating behaviors over time. An obvious choice is a PC-based control system as it is the most flexible one. In section 2, a control system which is based on PC-104 will be presented. In section 3, the structure of control program which equips the robot with suitable behaviors for playing soccer will be explained. This program was written to run on RT-Linux operating system. Some conclusions will be made in section 4.

2. Hardware architecture.

Specification

Degree of Freedoms	18
Height	495mm
Width	150mm
Weight	2.5kg

Control system

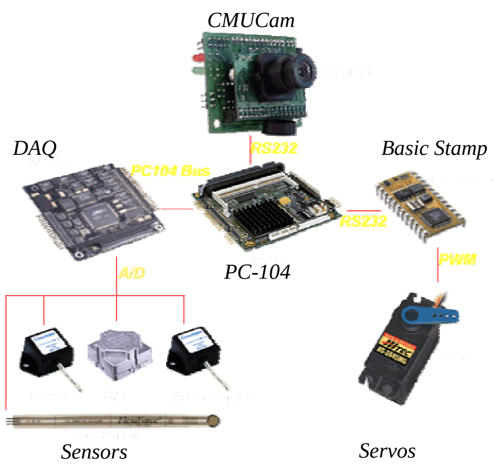

Fig. 2. Hardware architecture

Computer system

- ***PC-104:*** An ultra-compact size PC (3.6" x 3.8"), which is referred to as PC-104, is used as central processor of the control system. With this form factor, we can put an entire PC system in the robot's body and make use of its computation power and programming flexibility.
- ***DAQ card:*** Signals from various sensors will be collected by this data acquisition card. This DAQ card from Diamond System Inc communicates with the PC through PCI bus and provides more IO ports than needed with up to 32 analog inputs, 24 digital I/Os and 4 analog outputs. Thus, we can expand our system with more sensors in future without any problem.
- ***Basic Stamp:*** With the DAQ card, we are completely able to generate up to 18 PWM signals to control 18 servomotors. However, this task requires extensive processing time and leaves little time for control algorithm computation. The best solution is using a microcontroller to generate the PWM signals and fully utilizing the processor for higher level processing. Basic Stamp is a popular microcontroller that can perform this task perfectly. It communicates with the PC through RS-232 serial port at maximum speed of 115200 bps.
- ***Operating system:*** As any other PC system, this control system needs an OS. And more than that, a real-time OS is required because control action needs to be performed strictly every certain period of time. Failing to do so means a system crash. Having considered many criteria, we have selected RT-Linux OS. This real-time OS is free and sufficient for our application with worst-case latency of 30µm.

Sensors

There are several sensors needed to make the biped walk stably. These include:

- ***Force sensor:*** Force sensors are used to sense the center of pressure on the feet sole. We are using FlexiForces with measurement range of 25lbs.
- ***Accelerometer:*** This 3-axis accelerometer is a product of Crossbow Inc. It is used to sense the acceleration of the robot. It can also be used to derive translational velocities of the body.
- ***Gyro:*** This sensor of Silicon Sensing System is used to sense the direction in which the biped is heading. Actually this sensor gives information about angular velocity, and we need to integrate it to retrieve absolute angle information.
- ***Tilt sensor***: A dual axis tilt sensor from Crossbow Inc is tested to sense the tilt angles of the body. From experiment we found that this sensor is affected by acceleration. As an alternative, this sensor was replaced by two rate gyros.
- ***Camera:*** This is the vision sensor of the robot. We are using CMUCam which is a low cost yet powerful camera with some excellent built-in image processing functions.

Actuators
Actuators are also a very important element of a biped. A perfect actuator should be a light-weight yet high-torque motor. We have found that servomotors, which are used extensively in hobby aeroplane, are very suitable for this application. Initially we used Futaba motor, but it is not strong enough with a stall torque of 8 kgcm. We are now using Hitec motor with torque up to 13 kgcm.

3. Control program architecture.

As depicted in figure 3, in RT-Linux a control program usually consists of two parts: user-space programs and real-time modules. The user-space program is usually used to perform tasks which are not critical to the performance of the biped. On the contrary, real-time modules directly control the motion of the biped, so it has a huge impact on the performance of the biped. In RT-Linux, the two parts are separate. They share information through either FIFOs or mbuff shared memory. In our program, mbuff has been used because it is simpler to form bi-directional communication. If we want to use FIFOs, a pair of FIFOs must be established because each FIFO is uni-directional.

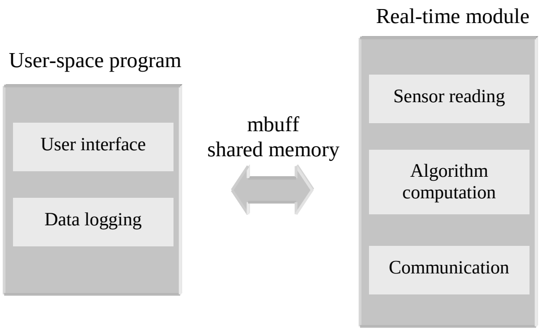

Fig. 3. Control program structure

User-space program. The tasks performed by this program have lower priorities than real-time tasks. In this program, we have a user-interface routine and a data logging routine. This user interface will prompt the user for inputting commands. After receiving the commands, it sends them to the real-time modules for realization. The data logging routine collects data from the real-time modules and stores it into files so that we can analyze the performance of the biped later on. This program absolutely does not need real-time behavior.

Real-time Modules. The robustness and stability of the biped are decided by these modules. They must perform at least three real-time tasks: collect sensory data, perform control action computation and communicate with low-level controller. The data need to collect include signal from three rate gyroscopes, one 3-axis accelerometer, eight force sensors and one camera. After collecting raw information, the sensor reading routine has to perform signal processing to derive necessary information. The sampling rate for each sensory processing task can set differently. For example, the sampling rate of gyroscopes and accelerometer signals may be higher than that of force sensors and the camera. This is because signals from gyroscopes and accelerometer must be integrated before using, while signals from force sensors and camera can be used directly. Variation in sampling rate can reduce the burden of computation for the main processor significantly.

Information perceived by the sensor reading routine will be used for control algorithm computation. The control action produced by the algorithm computation will then be sent to the low-level controller which directly controls the motion of each actuator. In our system, the low-level controller is Basic Stamp BS2p24. This controller needs to be provided with joints’ position in terms of duty cycle of PWM signals every 20ms.

4. Conclusion.

With a control system built based on standard PC architecture, we have created a platform facilitating the development of humanoid robot. With this platform, it will be simpler and more flexible to do research on several issues related to bipedal control and artificial intelligence.

Using this platform, we have implemented some soccer playing skills in our humanoid robot to participate in RoboCup 2004. It is now able to walk stably, approach and kick a ball. Other interesting behaviors are being implemented.