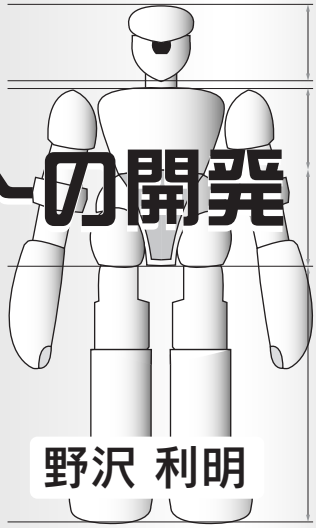
ネット経由でモーション変更可能な二足歩行ロボットの開発



第4回 新ロボット制御ライブラリOpenRoads-nerveの概要

二足歩行ロボットSPC-101の動作プログラムを作成するに当たり，本連載ではこれまでライブラリOpenRoads-bossを使用していた．今回はこれを拡張したライブラリであるOpenRoads-nerveについて解説を行う．（編集部）

野沢 利明

前回までは OpenRoads-boss を使ったアプリケーションの解説でしたが，いかがでしょうか．意外と簡単にロボットが動くのですが，興味を持っていただけましたか?

OpenRoads-boss ではモーション定義ファイルからの動作を前提にしているため，モーション定義ファイルを動的に作ってロボットへ送信しないと細かい制御ができません．また，基本的にロボットからの入力をサポートしていないため，5月号掲載の立命館大学の小川研究室で作成されたアプリケーション（pp.148-154）では，ロボットからの入力部分でかなり工夫されたようです．

今回は，OpenRoads-boss ではサポートしていない機能を拡充し，サーボなどのハードウェアの個別制御，モーション実行能力向上などを目的とした，最もハードウェア寄りの層になる OpenRoads-nerve を紹介します．

OpenRoads-nerve は現時点で開発中であり，製品としてのリリースはできていません．しかし，「使ってみたい/使えるならロボットを購入する」という要望が多ければ提供する手段を考えたいと思います．

これは今後，スピーシーズ製ロボットのベースになるシステムであり，6月19日に発表した次期システムでも採用しています．ぜひ，皆さんも使ってみてください．利用になりたい場合はシステムの入れ替えが必要で，SPC-101 の miniSD の交換（有償）が必要となります．

1．OpenRoads-nerve でできること

OpenRoads-nerve で可能となることを以下に示します．

● サーボ制御

ON/OFF 制御が可能です．角度指定で動作させる場合には，移動時間も同時に指定します．

また，サーボ・ステータス読み込みも行えます．

● パラメータ変更 [注1]

コンプライアンス・マージン，コンプライアンス・スロープ，パンチのパラメータが変更可能です．

● LED 基板

LED 点滅の固定パターンを用意しました．

● モーション

タイミングを指定して，RS-485 デバイス（サーボ，LED 基板）を制御できます．この際のリアルタイム性を向上しました．また，タイミングに間に合うようにパケットを送れば後はロボット内の nerve サーバが面倒を見てくれます．ただし，RS-485 パケットを編集して送る必要があります．

● イベント検出

モーション実行中にもイベント検出が可能です．ただし，RS-485 の通信量が増えると，センサのポーリング周期を守ることが困難になります．現状では，モーション実行中はポーリングを行わない方がよいでしょう．

● スイッチ読み込み

胸ボタン，両手先ボタンのスイッチ読み込みが可能です．また，サーボ負荷検出（予定），転倒検出（加速度センサ）なども行えます．

改善された点は次の通りです．

● モーション実行のタイミング安定性が向上

ソフトウェア・レベルの性能向上でモーション実行のタイミング安定性が 200ms → 50ms へと向上しました [注2]．

注1：ただし，パラメータを変更するとサーボは保証対象外になる．

注2：タイミングの精度については，オシロスコープで厳密に測定しようと思いながら，現状ではまだ測定できていない．ただ，50ms の角度指示にサーボ内のモータが追従できていないように見えるので，SPC-101 としてはあまり重要ではないのかもしれない．

KEYWORD ── SPC-101，二足歩行ロボット，Java，OpenRoads-boss，OpenRoads-nerve

●タイミング安定性向上の結果，サーボ動作音が静かになった

タイミングがしっかりしたため，モータが全力で動くことが少なくなったからと思われます．

●ネットワーク・クライアントでは未対応だったサーボ・コマンドを実装

●胸，両手にあるボタンを使用可能にした

●ユーティリティ関数を用意

ロボットとの接続は，ホスト名（またはIPアドレス）を指定して関数コールするだけです．IPv6にも対応しています．

●サンプル・プログラムを用意

試験に使用している，xUnitコードをそのまま提供しています．接続先を合わせるだけで，一通りの機能が確認できます．

ただし，xUnitとしてCUT[注3]が必要です．FreeBSDではportsのdevel/cut，NetBSDではpkgsrcのdevel/cutをインストールしてください．

2．サンプル・プログラムの概要

● 必要な開発環境

現在のところ，サンプル・プログラムの実行環境はFreeBSDとNetBSDです．Javaのクラス・ライブラリも現在開発中です．

クライアント側のライブラリ・ソースは公開を予定しています．皆さんからの声があるとそれが後押しとなって，公開が早くなるかもしれません．

用意されているFreeBSD/NetBSD用ライブラリの仕様を**表1**に示します．

クロス開発環境構築は，

```
http://www.netbsd.org/docs/guide/en/
chap-build.html#chap-boot-cross
```

表1　FreeBSD/NetBSD用ライブラリの仕様

環　境	備　考
FreeBSD 6.x-RELEASE i386	パソコンなど，外部から制御する
NetBSD-3 i386	
NetBSD-2 evbsh3	ロボット内部に実装するためのもの．NetBSDのクロス開発環境の構築が必要となる

```
-compiling-kernel
```

を参考にしてください．具体的な手順は，かなり適当な英語ですが，

```
http://sourceforge.jp/projects/
openroads/wiki/ProgrammingEnvironment
```

に記述しています．企業在籍者で，「お金よりも時間を節約したい」という方には，スピーシーズからも有償で提供していますが，基本的には上記の環境構築でできるものと同じものです．

3．サンプル・プログラムの動作

サンプル・プログラムの中から重要なポイントを抜き出して解説します．

サンプル・プログラムは，

```
http://sourceforge.jp/projects/openroads/
files/
```

のnerve.client_sampleパッケージからダウンロードしてください．これを展開すると，client/nerveClient.c があるので，詳細はソースを参照してください．

今後，変更する可能性もあるので，その点はあらかじめご了承ください．

● 接続

```
int open_cli_sock(ホスト名, ポート番号)
```

ホスト名とポート番号を指定して，ロボット内のサーバに接続します．ホスト名の代わりにIPアドレスを指定することもできます．ポート番号のデフォルト値は6809です．

戻り値は，connect(2)したソケットの記述子となり，以降のリクエストやnerveサーバからのレスポンス受信に使用します．

● リクエスト送信/レスポンス受信

パケット送受信用構造体群を nerveMain.h で定義しています．

readPacket()でnerveサーバからのレスポンスを受信します．

writePacket()がリクエスト送信用の汎用関数です

注3：http://www.falvotech.com/content/cut/