Tiny Konoha: ET ロボコン向けのスクリプト処理系の簡素化

志田 駿介†
菅谷 みどり†
倉光 君郎†

ET ロボコンは，2005 年から開催されている組み込みソフトウェアのロボットコンテストである．与えられたロボット筐体の上で，モデル記述とソフトウェア開発を競うことを目的としている．近年，TOPPERS OS など，ソフトウェア開発環境の整備も進められている．我々はスクリプトの利用により，競技会やチューニングの際のプログラムのコンパイルやロード時間を短縮することができると考え，開発している Konoha 言語のスクリプトの簡素化を目標とした Tiny Konoha を提案する．Tiny Konoha による省メモリ化の結果，NXT のハードウェア上で動作する実用的な実行環境を提供することに成功した．論文ではこの取り組みの経緯と簡素化のための設計,実装及び評価を述べる.

Tiny Konoha: Downsizing a Script Language for ET RoboCon Hardware

SHUNSUKE SHIDA
MIDORI SUGAYA
KIMIO KURAMITSU

ET RoboCon is a robot contest for embedded system software that has been held since 2005. The purpose of this contest is to develop a competitive software and model that achieves good performance by using a given robot hardware. These days, the software environment such as TOPPERS OS is also developed as a base system of the robot. On this robot, we consider a scripting language can reduce the compiling and loading time that are statically linked with OS at tuning time. Based on this idea, we have developed Tiny Konoha based on our developed scripting language Konoha Script, and successfully achieved the purpose of a practical runtime environment that can running on the NXT hardware. In this paper, we describe the history of our effort, and design and evaluation of our developed system for downsizing of Tiny Konoha.

1. はじめに

近年，組み込みシステムにおけるソフトウェア開発の手法としても，スクリプト言語の活用が始まっている．本稿では，ET ロボコン競技会の公式ハードウェアである Mindstorms NXT[1]上にスクリプト言語処理系を搭載する試みを行った．技術課題は，スクリプト言語処理系のコンパクト化の実現である． 我々は，静的スクリプト言語として Konoha の設計と開発を行ってきた．Konoha は，もともと組み込みシステム上の応用も想定され，たとえば，静的に型付けされた独自設計のバイトコードは，バーチャルマシンからの動的型検査の除去と，さらに CPU 資源の少ないマシンでも良好な実行性能を実現することができた．

一方，スクリプト処理系は，コーディングとプログラム実行の一体化された開発サイクルを実現するため，開発環境（パーサからコード生成）と実行環境（言語ランタイムとバーチャルマシン）の統合が必要であった．開発環境と実行環境が分離した Java VM とは異なる．そのため，開発環境の分だけランタイムが大きくなり，我々の Konoha 開発経験では，最小限の構成にしてもバイナリサイズで 100KB 以下を実現することはできなかった．このサイズでは，明らかに 64KB の NXT 上で動作させ

†横浜国立大学

ⓒ 2012 Information Processing Society of Japan

ることはできない．

本研究の目的は，ET ロボコン向けのスクリプト処理系の簡素化である．スクリプトの動作により，競技会やチューニングの際のコンパイルやロード時間を短縮することができる．目的の実現のために，本研究では Tiny Konoha の設計および実装を行った．Tiny Konoha は，Konoha とソースコードレベルの互換性を維持しながら，より極小な言語ランタイム，バーチャルマシン，ガベージコレクションを実現している． TinyKonoha では，コンパイラと実行処理系の分離と Bluetooth 通信によるバイトコードの転送などメモリサイズの削減を行うことで，メモリ 64KB での動作を可能としている．

本発表では，Tiny Konoha の設計、メモリ削減の具体的な手法、スクリプト言語を用いた組み込みシステムの開発例を示す。

2. スクリプトによる組み込み開発

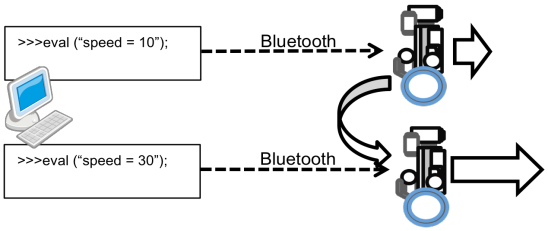

図 1 スクリプトによるインタラクティブ操作

本発表の目標は Tiny Konoha で効率的な組み込みシステム開発を行うことである．図 1 のように，ターミナル上から入力したスクリプトをその場でコンパイルし，NXT 上で実行することでインタラクティブな操作の実現を目指している．

現在はその事前段階として，スクリプトをターミナル上でコンパイルし，NXT に備わっている Bluetooth 通信機能を用いてコンパイル済みのバイトコードを転送し，実行する機能を構築している．この機能を用いることで，クロスコンパイルと組み込み機器への実行コードの転送という手間を省くことができる．

3. スクリプトの実行とリアルタイム性

我々は組み込み機器の制御を行う際に求められるリアルタイム性について，Tiny Konoha のスクリプトを NXT 上で 4ms 周期で動作させることで評価を行った．

Tiny Konoha でライントレースを行うためのスクリプトを図 2 に示す．NXT 走行体下部に取り付けられている光センサの値を読み込み，ラインの上かの判定を行う．ライン上と判定された場合には右に旋回を行いながら直進し，ラインをはみ出た場合は逆に旋回を行いながら直進する．これを繰り返すことでラインに沿って NXT 走行体を走らせることができる．System.balanceControl 関数は倒立振子を行うための関数であり，NXT の走行スピードと向きを引数に渡す．このようなリアルタイム性を求められるスクリプトに GC が与える影響を考慮するため，プログラム中で文字列の生成を行い，意図的に GC を発生させている．

```
void mainLoop () {
  ecrobotSetLightSensorActive();
  while (ecrobotIsRunning()) {
    manipulateTail();
    int turn = 0;
    if (ecrobotGetLightSensor() >= 600 /* threshold */) {
      turn = 50;
    } else {
      turn = -50;
    }
    new String(" ");
    balanceControl(50, turn);
    waiSem(); // wait 4ms
  }
}
mainLoop();
```

図 2 サンプルスクリプト

本スクリプトで一定時間 NXT を走行させ，balanceControl 関数を呼び出す間隔を 4ms 以内に呼び出すことができているか調査を行った結果を表 1 に示した．NXT に搭載されている CPU の周波数は 55MHz であり，時間は API から得られる最小単位である ms で表している．

ループ回数	GC 回数	平均 GC 時間	4ms 超過回数
100000 回	1428 回	1ms 以下	0 回

Tiny Konoha のヒープ領域は NXT の資源制約から 4KB となっている．現在の GC アルゴリズムは MSGC であるため最大停止時間が問題になる可能性もあったが，オブジェクトの探索がヒープ領域の小ささから短時間で終了しているため，1ms 以内で GC が終了している．

参考文献

[1] レゴマインドストーム公式サイト
http://www.legoeducation.com/mindstorms

[2] 白川 洋充，竹垣 盛一，システム制御情報学会編:リアルタイムシステムとその応用， システム制御情報ライブラリー,朝倉書店, 2001

ⓒ 2012 Information Processing Society of Japan