研究速報

RT-Linux に基づく視覚サーボシステムの開発

橋 本 浩 一* 則 次 俊 郎*

An Implementation of Visual Servo System on RT-Linux

Koichi Hashimoto* and Toshiro Noritsugu*¹

This paper describes the implementation issues of visual servo controller on a Linux-based real-time operating system. The Linux operating system is a Unix-like multi-user multi-task operating system and it is distributed freely. The real-time linux module adds a capability for hard real-time functionality, that is necessary for robot control, to Linux. Also a graphical user interface has been implemented on the basis of X Window System and Qt library. The implementation issues and some experimental data are presented. Discussion on the real-time control against the disturbance loads such as optical flow computation and heavy background processes are given.

Key Words: Real-time Controller, Linux, X Window System, Visual Servoing

1. は じ め に

ロボットの知能化において、種類の異なる複数の外界センサの信号をフィードバックすることは極めて重要である。エンコーダ・カウンタ、AD/DA 変換器などの応答の速いセンサの場合にはタイマなどを用いて定期的に入出力 (I/O) を行うプログラム制御型が有効である。しかし、応答の遅いあるいは処理が不定期なセンサの場合には、センサの信号が利用可能になったというイベントを割り込み信号としてコントローラに知らせ、コントローラで割り込み処理を行う割り込み制御型が必要となる。リアルタイム処理とは、外部からの非同期イベントに対して、ある定められた応答時間の制約のもとで行う処理をいうが、割り込み制御型の I/O を用いる場合には、このリアルタイム性が重要になる。

ロボット研究においては、割り込み応答性能を重視するため、オペレーティングシステム (OS) として VxWorks や pSOS などのリアルタイム OS または DOS が主流であった。しかし、リアルタイム OS は一般に高価で、研究室の実験用としては負担が大きい。また、最近の周辺装置 (デジタル入出力ボード、ネットワークボード、画像処理ボードなど) では DOS のドライバは提供されず、DOS の実験環境の維持も困難になってきた。

一方、Windows (Windows95, Windows98, WindowsNT など) では豊富で安価な周辺機器が利用できるため、研究室の実験には好適である。しかし、リアルタイム処理を行う場合、32 ビット系ではユーザアプリケーションから割り込みを制御

することは不可能で、割り込み制御を行うためにはハードウェア固有の仮想デバイスドライバが必要になる。この仮想デバイスドライバ開発ツールは通常のアプリケーション開発キットには含まれておらず、別途購入しなければならない。また、PC-Unix (Linux や FreeBSD など) も豊富な周辺装置が利用でき、OS に加えてエディタ、コンパイラ、デバッガ、グラフィカルユーザインタフェース (GUI)、ネットワーク機能など、開発環境は無償で入手・使用できるので経費は低く押さえられるが、通常の Unix ではリアルタイム処理は不可能である。

Real-time Linux (RT-Linux) [1] [4] は、Linux にリアルタイムカーネルを付け加えるためのモジュールである。RT-Linux では、Linux カーネルは最も優先度の低いリアルタイム (RT) プロセスとしてスケジューラされ、RT プロセスが CPU を要求するとそれに CPU を横取りされる。このようなスケジューラを Linux に付け加えることにより Linux の特長を生かしたリアルタイムシステムが構築できる。

本論文では、RT-Linux に基づく視覚サーボシステムの開発を通し、その利点と問題点を議論する。RT-Linux を用いたロボットシステムはすでに報告されているが [5]、本論文では割り込み処理に注目し、ユーザ操作の利便性とリアルタイム性の両立を目指した RT-Linux の利用法を検討する。具体的には、富士通トラックビジョンボード [10] と三菱電機ムーブマスタを用いた視覚サーボシステムを構成するにあたり、処理をリアルタイム性を必要とする部分とそうでない部分にわけ、それぞれを RT プロセスと Linux プロセスとして実装する。そしてこれらのプロセスの間の通信を RT-FIFO によって行うことにより操作性の良いユーザインタフェースとリアルタイム処理を同

原稿受付 1998 年 5 月 14 日

*岡山大学工学部

*Faculty of Engineering, Okayama University

時に実現する方法について述べる。

2. システムの概要

ムーブマスタのリアルタイムインタフェースは、パーソナルコンピュータ (PC) と Arcnet で接続される。ロボットは制御周期 28.5 [ms] ごとに Arcnet 経由で PC に割り込みをかけ、関節角データを送信する。送信後 3 [ms] 以内に PC から目標関節角データを受け取れば、ロボットはその制御周期で目標関節角に移動するように関節角サーボ制御される。3 [ms] 以内に受け取らなかった場合は、ロボットはその制御周期では現在位置を保持する。つまり、3 [ms] のリアルタイム処理を実現できない場合には、ロボットは移動と停止を繰り返し、極めてぎこちない動作となる。したがって、この「目標関節角の送信抜け」(制御抜け) は必ず回避しなければならない [8]。このリアルタイム要求を満たすために、本システムではロボットからの割り込み処理に RT-Linux を用いる。Arcnet の割り込みハンドラは RT プロセスとして実装する。割り込みハンドラでは、上位のビジョンプロセスからの特徴点偏差の受け取り、現在の関節角の読みだし、関節角指令の計算、そしてロボットへの指令値の送信などを行う。

ビジョンプロセスは (リアルタイムではない) Linux プロセスとする。これは、トラッキングビジョンドライバ [9] が RT-Linux に対応していないためであるが、以下の条件を考慮にいと、この選択は妥当である。

- (1) 関節角サーボのリアルタイム要求の方がきびしい … 関節角サーボのリアルタイム性が失われると、ロボットの動作がぎこちなくなるだけでなく、ロボットのモータやギアにも悪影響を及ぼす。
- (2) RT プロセスはできるだけ小さく保たなければならない … 操作性を良くするためには、RT プロセスを小さくし、Linux プロセスに実行の機会を与える必要がある。
- (3) 画像処理時間は視覚サーボに用いる特徴量に依存する … 例えばビデオフレーム同期信号の立ち上がりでビジョンプロセスを定期的に起動することにする。このとき特徴点数を増加させると画像処理時間が起動周期を上回り、Linux プロセスは実行の機会を奪われる (ハンガアップ)。

関節角サーボ系への指令は視覚サーボ法により生成する。視覚処理のサンプリング周期が 33.3 [ms] の場合、およそ 3 秒に一度の割合で一つの視覚フィードバックサイクルに二回の関節角サーボ割り込みが入り、視覚サーボ指令が欠落する。したがって、RT プロセスにおいて特徴点偏差または関節角指令を適切に外挿する必要がある。一方、割り込みハンドラに外挿機能を与えると、上位のビジョンプロセスとの結合に柔軟性が生じる。言い換えると、ビジョンプロセスの定期性が保証されなくなったとしても、制御抜け (目標関節角をリアルタイムに送信できずにロボットが急停止すること) を避けることができる。ビジョンプロセスを Linux プロセスとすることで、RT プロセスは非常に軽い割り込み処理 (関節角目標値生成と外挿) のみとなる。したがって、Linux プロセスの実行機会が増え、GUI 機能を用いて操作性を高めることも可能である。トラッキング

ビジョンのパラメータ設定、特徴点設定、フィードバックゲイン変更などの処理は GUI を用いることで格段に操作性が良くなる (画面上の領域を指定するのにカーソルキーとマウスでは操作性の差は極めて大きい)。本研究では Qt ライブラリを用いて GUI を構成する。

3. 視 覚 サ ー ボ

フィードバックループに視覚センサを組み込んだ制御系は視覚サーボ系と呼ばれる [6]。Fig. 1 に視覚サーボ系の例を示す。カメラからの出力を画像処理し、特徴量のみをフィードバックする視覚サーボ法を特徴ベース法と呼ぶ。特徴ベース法において特徴量の選択は制御系の性能に大きく影響する。この関係を明らかにするために筆者らは視覚サーボ系における感度を定義し、感度の向上が精度の向上、反応速度の向上、そしてノイズの影響の低減に効果があることを示した [7]。しかし、画像処理装置の速度の制限により十分な数の特徴点を用いて実験できなかった。そこで本研究では、視覚サーボ実験中でも GUI を用いて容易に特徴点を追加、削除可能なシステムを構成する。

3.1 RT-Linux

Linux をリアルタイムシステムとして用いるにはいくつかの障害がある。最も大きいのは、カーネルの実行時には割り込みが禁止されていることで、そのほかにはプロセススケジューリングの不均一さ、プロセススイッチの粗さ、高分解能のタイマーの欠如などがあげられる [1]。RT-Linux は Linux に RT カーネルを付け加えたもので、RT プロセスは RT カーネルと同じ特権レベルを持ち、デバイスに直接アクセスすることができる。また、Linux カーネルは最も優先度の低い RT プロセスとなる。タイマや外部割り込みによって RT プロセスが周期的に実行される場合、周期内に RT プロセスが終了しなければすべての CPU 資源が RT プロセスに使われてしまい、Linux カーネルは実行の機会を与えられない (キーボードやマウスの入力も受け付けなくなる)。したがって本当にリアルタイム処理が必要な最小限の処理のみを RT プロセスとし、残りをすべて Linux プロセスとして記述しなければならない。

RT プロセスと Linux プロセスは RT-FIFO (First In First Out バッファ) という専用の I/O インタフェースを通して通信する。Linux プロセスからは RT-FIFO は通常のキャラクター

Fig. 1 Visual servoing

Fig. 2 Real-time FIFO

バイスとして取り扱うことができる (Fig. 2). RT プロセスはカーネルモジュールとして記述し、専用の関数で RT-FIFO に入出力する. Linux ではリポートせずにカーネルモジュールをロードすることができるので、実験前に RT-FIFO, RT スケジューラと RT プロセスをロードし、実験後にアンロードすれば、実験以外のときは通常の Linux として利用できる.

3.2 Qt ライブラリ

Qt ライブラリは C++ を用いて GUI を開発するためのオブジェクト指向の開発環境ライブラリであり, X 上の無償ソフトウェアの開発に用いる限りソースが無償で提供されている [2]. Qt では低レベルの Motif/X11 のコードなどを書く必要がないので開発が効率的である. さらに, オブジェクト間の通信にシグナル・スロット機構を備えており, 様々な形態の通信・同期が可能になる.

3.3 トラッキングビジョン

富士通カラートラッキングビジョン (PCI 版) TRV-CPW5 は, 専用ハードウェアを用いて相関演算を高速に実行することによりビデオレートでの追跡処理ができるボードである [10]. トラッキングビジョンのドライバは, 富士通からは Windows95 用のみしか提供されていないが, Linux 用ドライバが東大で開発され公開されている [9]. ただし RT-Linux に対応していないためビジョンプロセスは Linux プロセスとする.

4. 実 験

画像処理を Linux プロセスとして実装することにより, Qt を用いた GUI を利用することができ, マウスを用いた快適な操作性が実現できる. また, ロボットを動かしながらサーボ系のゲインを調整したり, ロボットの軌道を画面に表示することも可能になり, 格段に利便性が向上する. この環境下におけるリアルタイム性について考察する.

4.1 実験装置

RV-E2 の手先に CCD カメラを取り付け, オブジェクト画像上に数個から数十個の追跡ウインドウを設定し, 画面上でのウインドウ位置が初期設定位置に戻るようによりロボットを制御する (Fig. 1). RV-E2 のリアルタイムインタフェース (2E-32ARC) と PC (Fujitsu FMV-D6200, CPU: PentiumPro 200 [MHz], メモリ: 64 [MB]) は Arcnet により接続される. Arcnet ボー

ドはアドバネット製 AISA1517A (ISA) で, ドライバは 2E-32ARC 付属の DOS 版ドライバを RT-Linux 用に書き換えた. ロボットコントローラから 28.5 [ms] ごとに Arcnet を通して PC に位置要求割り込みが入り, 割り込みハンドラは現在の関節角の読み出し, 目標関節角の計算, 指令値の送信を行う.

Linux プロセスにおいて, トラッキングビジョンのサンプリング周期より若干短い周期で特徴点抽出関数を起動する. その関数内でビデオのフレーム同期信号を待つので, 実際は 33.3 [ms] の整数倍の時間間隔で各追跡ウインドウの x, y 座標が読み出され, RT-FIFO に書き込まれる. RT-Linux では RT-FIFO にハンドラを設定し, FIFO にデータが書き込まれると直ちにそれを読み出す. また, 関節角変位のみではなく, RT プロセスの制御のための通信も RT-FIFO を通して行う.

4.2 実験結果

まず, 追跡ウインドウ (ウインドウサイズ: 64×64 [pixel²], 間引き 4 倍, テンプレートサイズ: 16×16 [pixel²]) を 10 個選び, 視覚サーボ実験を行った. バックグラウンドで起動されている Linux プロセスは, 通常のデーモン群と三つの kterm, 一つの emacs と一つの netscape である. このときの視覚サーボループのサンプリング時間を Fig. 3 に示す. 1 秒付近と 10 秒付近で画像処理関数が待たされているが, これは 1 秒付近においてマウスでゲイン変更ウインドウをポップアップさせ, 10 秒付近においてゲインを変更したためである.

次に, カメラを手先からはずし三脚に固定して, 400 点のオプティカルフローの計算を行い, そのフローの平均の方向によりロボットを動かす実験を行った. オプティカルフローはモノクロでウインドウのサイズは 8×8 [pixel²] とした. つまり, 基本相関演算は 400 回である. また, この実験の途中で emacs や netscape などを新たに起動し, Linux プロセスの負荷をあげてみた. このときの視覚フィードバックのサンプリング周期を Fig. 4 に示す. 負荷が大きいとサンプリング周期は一定せず, ロボットの動作速度も一定ではなかった.

最後に, RT プロセスの応答の周期性を調べた. オプティカルフロー実験のときの結果を Fig. 5 に示す. およそ 0.5 [ms] のばらつきはあるものの, 割り込みは周期的に処理されていることが分かる. また, 制御抜け (割り込み応答が 3 [ms] 以上遅れてロボットの移動が 0 になること) も生じていない. 本研究では RT-Linux のタイマで計測しているため割り込みに対する応答遅れ (割り込み信号が入ってから割り込みハンドラが起動されるまでの遅れ時間) は計測できなかったが, 文献 [3] によると負荷によって変動するものの Pentium 120 [MHz] で 2.3–4.4 [μsec] である. また, RT プロセスの応答時間はシステムの負荷には関係なく, 上記の二つの実験を含む多数の実験においてほぼ同じ結果が得られた.

4.3 問題点

本システムでは, Linux プロセスで特徴点位置検出と RT-FIFO への書き込み, RT プロセスの FIFO ハンドラでそのデータの読み出し, ハードウェア割り込みハンドラで関節角変位計算とロボットコントローラへの送信を行う. したがって, 特徴点位置検出に 33.3 [ms] 以上と割り込み待ちで最大 28.5 [ms] の遅れが生じる. また, トラッキングビジョンプロセスを Linux

Fig. 3 Sampling time for 10 windows

Fig. 4 Sampling time for optical flow with disturbance

Fig. 5 Sampling time of RT task

プロセスとしているので、GUI 処理を含むバックグラウンドプロセスが多いときには Fig. 4 のような「コマ落ち」が生じる。

遅れの問題は、ビジョンおよびロボットコントローラのサンプリング周期が比較的長いことによるもので、通常のカメラとムーブマスタロボットを用いる以上避けられない。しかし、コマ落ちの問題は、トラッキングビジョンライブラリを RT-Linux 対応にしていないことに起因するので、ドライバの改良により改善できる可能性がある。また、GUI 処理のために生じる画像処理タスク待ちは、画像処理タスクと GUI 処理を一つの Linux プロセスとして生成したためである。この問題はこれらの処理を別プロセスとして優先度に差を持たせることにより改善可能である。

5. お わ り に

本論文では、Linux, RT-Linux, Qt ライブラリを用いて PC ベースの GUI をもつロボット制御環境を構築する例を示した。RT-Linux のハードウェア割り込みに対する応答性は RV-E2 の制御には十分であることが確認できた。また、Qt を RT-Linux と併用した GUI 環境はリアルタイム性と利便性を両立することが確認できた。

謝 辞 本研究は平成 9 年度文部省科学研究費補助金重点領域研究 (1) とメカトロニクス技術高度化財団研究助成の援助を受けた研究の一部である。

参 考 文 献

- [1] M. Barabanov: "A linux-based real-time operating system," Master Thesis, New Mexico Institute of Mining and Technology, 1997.
- [2] Troll Tech: Qt documentation, <http://www.troll.no/qt/>, 1998.
- [3] A.L. Veiga: Testing RT-LINUX, Laboratorio de Electronica Report, Physics Department, La Plata University, Argentina. <http://beba.fisica.unlp.edu.ar/rt.html>, 1998.
- [4] V. Yodaiken: The rt-linux approach to hard real-time, <http://rtlinux.cs.nmt.edu/>, 1998.
- [5] 加賀美聡他: "脚型ロボット感覚行動統合研究プラットフォーム JROB-1", 日本ロボット学会誌, vol.16, no.5, pp.623-628, 1998.
- [6] 橋本浩一: "視覚フィードバック制御—静から動へ", システム/制御/情報, vol.38, no.12, pp.659-665, 1994.
- [7] 橋本浩一, 青木篤人, 則次俊郎: "冗長な特徴量に基づく視覚サーボ", 日本ロボット学会誌, vol.16, no.3, pp.384-390, 1998.
- [8] 三菱電機: リアルタイムインタフェース取扱説明書, 1996.
- [9] 松本吉央, 坂井克弘, 稲邑哲也, 稲葉雅幸, 井上博允: "PC ベースのハイパーマシーン: 知能ロボットの汎用カーネル", 第 15 回日本ロボット学会学術講演会予稿集, pp.979-980, 1997.
- [10] 森田俊彦, 沢崎直之, 内山隆, 佐藤雅彦: "カラートラッキングビジョン", 第 14 回日本ロボット学会学術講演会予稿集, pp.279-280, 1996.