

# オープンソースソフトウェアの活用 —LinuxによるJava画像処理システム—

関東職業能力開発大学校 加部 通明

An Application of Open Source Software  
—Java Image Processing System on Linux—

Michiaki KABE

**要約** 昨今、パソコンのクライアントOSの95%を占めるマイクロソフトのWindowsに  
対抗して、LinuxをクライアントOSに採用する動きがLinspireやHPなど一部の  
企業により徐々に実現されてきた。すでに、サーバOSではその流れが国や自治体および  
ヨーロッパにおいて経費節減や独自システムの構築という観点から、近々30%位の占有率を  
確保する見通しである。Linuxを初めとするソースコードが公開されている所謂オープンソー  
スソフトウェア（Open Source Software 以下OSSと略記する）と呼ばれるソフトウェアは、自由  
に使用できて開発費や運用費をかなり削減でき、今後のシステム開発の活用にお  
いて大いに注目されている。

従来、画像処理システムはパソコンOSの一極集中性とドライバの関係からWindows上で  
CやVCと言ったC言語系又はVBによって開発されることが多かった。そうした中で、世の中  
の流れを先取りすべくOSSを活用して画像処理システムを構築することは、応用課程の開  
発課題実習において企業から提示された課題を開発する場合等に、まさしく打ってつけな予  
算削減手段であり有意義であると思われるので、本稿をOSS活用の一例として報告する。

## I はじめに

近年「情報格差」という言葉が差別用語の一種とし  
てしばしばマスコミに登場している。その意味は国や  
地域における情報の利用を上手く享受できているか否  
かを表した言葉で、経済格差がそっくりそのまま情報  
にも当て嵌まり、地域間格差を新たに生む火種にもなっ  
ている。余りが付かない簡単な例として、有償のソ  
フトウェアについて取り上げてみよう。

マイクロソフトのパソコンOSやその上で稼働する  
様々なソフトは1万円前後する。それを貧しい国でも  
同じ価格で売れば、当然買う人は一部に限られその国  
の大部分の人達は現在のインターネットを初めとする  
情報技術の恩恵を受けることができない。ハードウェ  
アはいろいろと中古があり、自動車と同じで部品を取  
り替えたりして改良すれば、ある程度の能力をもった

パソコンを安く手に入れることができるが、ソフトウェ  
アはそうはいかない。ハード一つに対して必ず新しい  
ソフトを先進国と同じ値段で購入しなければならない  
からである。このようなところから、貧しい国では未  
だに海賊版と言われる違法コピーが市場に出回って  
いるのである。

こうした情報格差を解消する一つ的手段として、  
OSSの活用が上げられる。無償だから全く価格の上  
からは問題ない。ではこのありがたいOSSが何故余  
り利用されていないのか、その理由を考えてみよう。  
それらを列挙すると次のようになる。

- ① どのようなOSSがあるのか分からない。
- ② OSSの利用方法が分からない。
- ③ 日本語処理対応がなされていないか、又は、不十分であって使いづらい。

今回、OSSの活用例を特にJNI（Java Native

Interface) に焦点を当てて説明し、更に、全てOSSを利用した画像処理システムの構築について報告する。

## II OSSについて

まず初めに、OSSの大まかな定義をここで確認しておく。定義の詳細は参考文献(1)で公開されている。そのサイトではOSSに関する歴史や背景、更にOSSの見つけたなども説明してある。

定義：OSSとは、次のような各項目についてある配布条件を満たすソフトウェアである(注1)。

- ① 自由な再配布
- ② ソースコードの開示
- ③ 派生ソースウェアの配布の許可
- ④ 作者のソースコードの完全性
- ⑤ 個人やグループに対する差別の禁止
- ⑥ 利用する分野への制限の禁止
- ⑦ ライセンスの分配
- ⑧ 特定製品でのみ有効なライセンスの禁止
- ⑨ 他ソフトウェアを制限するライセンスの禁止
- ⑩ ライセンスは技術中立でなければならない

ここで、あるWebサイト(2)が把握しているOSS情報からその現状と数量的推移を抜粋してみよう。2003年12月中旬と2004年11月中旬の11ヶ月間におけるOSS推移状況を表1に示す。これよりOSSが漸次増加傾向にあることと、計画中のものや試作品に類するものが数多く存在することが分かる。

表1 OSSの開発状況とその増減

	08/12 中旬	04/11 中旬	増減
計画中	13,783	14,462	+679
プレα	9,784	10,976	+1,192
α	9,239	11,042	+1,803
β	11,277	14,073	+2,796
製品安定	8,995	11,676	+2,681
成熟	882	1,052	+200
休止	376	922	+546

## III 画像処理システム

今日、画像処理はパソコンの演算能力の飛躍的な進歩によりいろいろな分野で応用されている。例えば、工場の生産現場では、従来人間が行っていた目視によ

る製品検査や欠陥検査等をコンピュータで自動的に行うようになって来ている。これらの検査は、従来ある単位で部品を抜き取りそれを検査してその単位での部品の良否を判定していたが、近頃では、小さな部品のビスでも全数検査という大変厳しい要求になっている。人間の目視では、検査する人の作業能力や時間の経過による検査精度の低下という不確実な要素を内包しているため、こうした検査をより確実に継続的に行うような自動検査システムの開発要求が従来からあった。

特にものづくりの支援を標榜する能力開発施設では、パソコンの周辺機器として機械制御や画像入力等を利用する機会が多い。以前はそれらをCプログラムで制御してきたが、今後はWebや通信、ビジネス関係等広範囲に対応できる汎用性のあるJavaプログラムを使用する機会が益々増えることが予想される。そうした状況の中で、画像処理システムをJavaで構築することは時代の流れに沿ったものと考えられる。

### 1 機能

表2に示したOSSを活用して表3に示すような画像処理機能を持ったシステムを構築した。

表2 利用するOSS.bmp

OSS	機能
Red Hat Linux 9	パソコンOS
Video for Linux two	ビデオキャプチャ用
btv.o	ビデオキャプチャボード用ドライバ
C	API使用言語
Java	画像処理言語

表3 処理機能

メニュー	処理項目	特別事項
メニュー	ファイル	処理のファイルから画像を読み取る
	カメラ(複数台)	Jpegによる画像を出力する
	カメラ(1台のみ)	401の処理開始により画像を表示する
	学習データ(32W)	32Wで撮影した学習データをトリックで入力する
	検査データ(32W)	32Wで撮影した検査データをトリックで入力する
前処理(カメラ)	画像ファイル	入力したファイルも画像認識済とする
	枚数	画像ファイルを保存する
	① 傾斜	傾斜修正機能はシステムを参考に傾斜を修正する
	輪郭抽出 Laplac	
	最大値	
	最小値	
	平均値	
	平滑化	
	反転	
	差分	差分抽出機能は差分を抽出する
画像処理(カメラ)	先鋭化	
	輪郭抽出 Sobel	
	輪郭抽出 Gabor	① 傾斜の傾斜修正パラメータを1で固定する
	傾斜修正(傾斜修正)	
	傾斜修正	カメラを傾斜修正後に傾斜修正プログラムを保持する
画像処理(カメラ)	傾斜修正	傾斜修正により傾斜修正を保持する
	傾斜修正	
	傾斜修正	
	傾斜修正	
画像処理(カメラ)	傾斜修正	① 傾斜修正の傾斜修正パラメータを1で固定する
	傾斜修正	傾斜修正プログラムを保持する
	傾斜修正	傾斜修正された傾斜の傾斜修正を抽出する
	傾斜修正	傾斜修正された傾斜の傾斜修正を抽出する
画像処理(カメラ)	学習	学習データを学習された傾斜修正する
	検査	検査データを学習に基づいて分類する

## 2 ハードウェア構成

職業能力開発大学校応用課程の授業で使用している CCDカメラと実勢価格約1万円という安価なビデオキャプチャを用いて<sup>(注2)</sup>、図1に示すような画像処理のハードウェアを構成した。このシステムでは、カメラから撮像した映像は直接パソコン上に表示されるので、特別なモニタは必要ない。

## 3 ソフトウェア構成

ソフトウェア構成を図2に示す。今回使用したソフトウェアは全てOSSを利用しているのでソフトウェアに関しては経費ゼロである。今後のシステム導入では、ハードウェア経費10に対してソフトウェア経費3と言う現状の高い割合をOSSの利活用で究極的には0に近い比率まで持っていくことが理想である。厳しい予算を余儀されることが予想される中で、各人が具体的な行動や対策として取ることができるのがOSSの利活用と考えられる。

## 4 画面設計

表3に示したようにメニューには、入力、前処理(フィルタ)、基本処理、パターン認識をそれぞれ設けた。このパターン認識はニューラルネットワークを利用し、対象物が特定されないあらゆる画像に対応できる意味で、究極の特徴抽出と考えられる濃淡画像の全画面を抽出した。これより、抽出画素数とメモリとの制約から150×150画素程度の画像を3種類しか分類することができない<sup>(注3)</sup>。

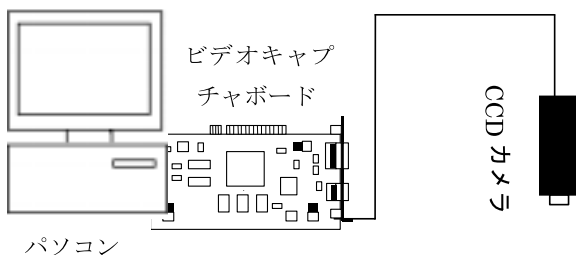


図1 ハードウェア構成

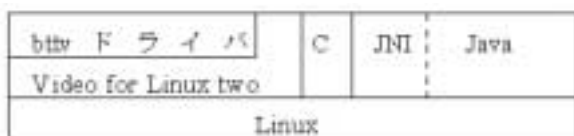


図2 ソフトウェア構成

## IV OSSの活用方法

ここでは、III章で説明した画像処理システムを取り上げて、ビデオキャプチャ<sup>(注4)</sup>選定から画像処理プログラムの作成と実行までのOSS活用方法を説明する。

### 1 Video for Linux two

Video for Linux two<sup>(3)</sup> (以下v4l2で表記する)は、ビデオキャプチャ用のドライバとカーネルとの情報の橋渡しをするモジュールvideodev.oと画像取得用プログラムを集めたAPIから構成されている。そして、それはLinux上でビデオキャプチャがカメラより画像を取得するために、そのAPIの関数をCプログラムに組み込んで利用される。

具体的には、v4l2の開発サイトからダウンロードして得られた圧縮ファイルを解凍し、その後makeしてモジュールvideodev.oを作成しそれを利用する。モジュール以外の全般的な利用方法を図3に示す。

なお、Linuxには既にカーネル2.2.x乃至2.4.xにおいて標準で最初からvideodev.oが組み込まれているので、それを利用してもよい。



図3 v4l2の利用方法

### 2 ビデオキャプチャの選定

画像処理システムを構築する上で一番厄介なのがビデオキャプチャの選定である。現在、国内ではv4l2用に開発されたドライバで稼働できるビデオキャプチャはほとんどが外国製であって、信頼性に幾分不安がある。以前販売されていたv4l2対応の画像処理チップBt848<sup>(注5)</sup>を使用していたアイ・オー・データ製のビデオキャプチャボードGV-VCP2/PCIが2002年に製造中止になり、その後継製品がv4l2に対応していないのでそれに代わる廉価な製品を探すのが難しい。

### 3 言語

C言語をv4l2を呼び出す制御用プログラムで使用し、Java言語をJNIを利用してその制御用プログラムを呼び出したり、GUIに優れていることから画像処理を行うプログラムで使用する。

#### 4 画像処理プログラム実行方法

全体の流れを図4に示し、以下細部を説明する。

##### 4.1 JNIを活用したv4l2による画像取得

JNI<sup>(4)</sup>はJavaと他の言語とを結び付ける道具である。Javaの短所にはドライバを直接操作することができない点がある。一般的に言うと、最近ではドライバのソースプログラムはC言語で書かれていることが多く、そのドライバを利用してプログラムを作成する場合にもC言語が適切である。今回利用したビデオキャプチャボードのドライバもC言語で使用する。そこで、JavaからCプログラムを呼び出すのがJNIの役割である。今までJNIを取り上げた例が少ないので、その具体的な利用方法を、今回の画像処理システムのカメラ撮像を例にして、ここで順番を追って説明する。

(1) 先ず、画像処理を行うメインのプログラムImageProv4l.javaを作成する。その中で、カメラ撮像を行う手続getImage4l()<sup>(注6)</sup>をオブジェクトimagev4lを用いて呼び出す文章を記述する。

```
imagev4l.getImage4l();
```

(2) (1)で作成したプログラムImageProv4l.javaをコンパイルする。

```
$javac ImageProv4l.java
```

(3) コンパイル後のクラスファイルImageProv4l.classを利用して、Cプログラムgetcamerav4l.cが使用するJNI用ヘッダファイルImageProv4l.hを作成する。

```
$javah ImageProv4l
```

(4) v4l2のAPIであるプログラムvlib.cとilib.cをmakeコマンドによりコンパイルして、目的プログラムvlib.oとilib.oをそれぞれ作成する。

(5) 画像取得用プログラムgetcamerav4l.cを作成して、makeコマンドによりコンパイルし目的プログラムgetcamerav4l.oを作成する。

(6) 3つの目的プログラムgetcamerav4l.o、vlib.o、ilib.oをmakeコマンドで結合して、共有ライブラリlibgetcamerav4l.soを作成する。

(7) 以上全てエラーなく終了した場合、videodev.oとドライバbttv.oをモジュールに組み込み、その後、画像表示プログラムxawtvを実行しビデオキャプチャボードおよびドライバが正常に動作しているかを確認する。

```
#modprobe bttv
#xawtv
```

エラーがある場合、途中にエラーメッセージが出力されていると思うので、それを参照し再度ドライバ等を組み込む。

(8) ビデオキャプチャボードおよびドライバの正常動作を確認した後、xawtvを終了させて画像処理プログラムImageProv4lを実行する。

```
$java ImageProv4l
```

以上の説明で使用したMakefileの内容を図5に示す。

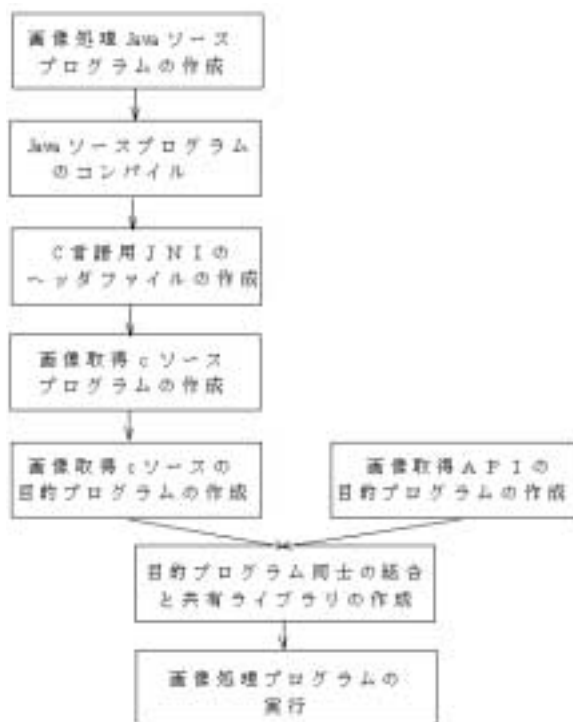


図4 プログラム作成と実行

```
#video4linux
CFLAGS=-Wall -O2 `imlib-config --cflags`
LFLAGS=`imlib-config --libs`
OBJECTS=vlib.o ilib.o
C2 = -shared
J1 = -Dusr/local/2sdct1.4/include
J2 = -Dusr/local/2sdct1.4/include/linux
#make shared library by linking object files
$(S)
    gcc -o lib$(S) so $(C2) $(OBJECTS) $(S) $(LFLAGS)
#make object file
.c.o
    gcc -c $(J1) $(J2) $(CFLAGS) $(S)
clean:
    rm -f $(OBJECTS) $(S)
```

図5 v4l2関連のMakefile

##### 4.2 v4l2について

カメラ撮像をパソコン画面に出力するv4l2の関数にipdisp()があるが、これを2回以上使用すると、図6に示すエラーメッセージが出てアボードしてしまうので、1回のみ使用に止める。もし、カメラ撮像を連









図16 PCI-5520によるJavaプログラムの画像表示

経費削減の具体的な対策の一つとして、本稿では、ビデオキャプチャの使用を前提として画像処理システムの構築を考えた。しかし、今後はそれを必要としないUSBカメラによる画像取得を行えば、それこそ数千円のカメラ代のみの経費でシステム構築が出来てしまう。USBカメラによる撮像の精度と動画の取り込みが出来るかと言う課題が解決されれば、後はv4l2等のOSSにその製品が対応しているか否かが問題となるだけである。

[注]

- (注1) ここで述べる定義はあくまでもOSSの推進を唱える団体によるものであって、ライセンスに係わる項目等は今後の社会情勢で変わる可能性がある。
- (注2) IV章1で説明するが、当初想定していたビデオキャプチャが製造中止になったので、現在では、中古店又はオークション等でしか手に入らない。
- (注3) 3層型バックプロパゲーションのニューラルネットワークを利用して、本稿のような手法でパターン認識を行う場合、プログラムの大部分を占める入力層と中間層間の結合係数のメモリ量が問題になる。今、記号M、N、T、wを以下のような意味とする。  
 M：パソコンのメモリリソースで約50メガバイト。  
 N：検査画像の一辺の画素数。(N×Nが画像の大きさであり、また入力層のユニット数ともなる。)  
 T：中間層のユニット数。  
 w：入力層と中間層間の結合係数で、具体的にはw[T][N][N]なるfloat型配列。  
 Javaではfloatは4バイトだから、配列wのメモリ量4×T×N×NはM未満となる。これを目安にパソコンで実験調査すると、おおよそN=150となった。
- (注4) 本稿では、ビデオキャプチャ製品名を製造元が使用している呼称を用いて表記している。

GV-VCP 2 /PCI：ビデオキャプチャボード  
 IP5005：画像処理ボード  
 PCI-5520：画像キャプチャボード

(注5) Bt848のドライバがbttv.oであり、それはvideodev.oを通してカーネルと情報のやり取りを行っている。従って、IV章4.1(7)で説明しているように、modprobe bttvを行うことによって、videodev.oとbttv.oの両方がモジュール組み込みされる。

(注6) Cプログラムのファイル名はgetcamerav 4 l.cであるが、その共有ライブラリのファイル名はlibgetcamerav 4 l.soとなる。Javaプログラムでその共有ライブラリを定義するときには

```
public static native byte [][] GetImagev4l();
static{
    System.loadLibrary("getcamerav4l");
}
```

として共有ライブラリ名をgetcamerav4lで表記する。実際にそのプログラムを呼び出す場合には、getcamerav4l.cの名前と関係ない名前が使えるGetImagev4l()とした。

(注7) IP5005では、カメラ撮像した画像やそれを処理した画像をパソコン画面に表示することができないので、別のモニタに表示する。もし、パソコン画面に画像を表示したい場合には、今回提案したようなシステム構築をしなければならない。

(注8) dpg0100.oは各カーネルバージョンにおけるAPI仕様や内部情報の違いを吸収する為のモジュールであり、cp5520.oがドライバ本体である。それらは、ちょうどvideodev.oとbttv.oとの関係と同じである。

(注9) IP5005及びPCI-5520の画像取得解像度はそれぞれ512×440、640×480である。

[参考文献]

- (1) <http://www.opensource.org/docs/definition.php>
- (2) [http://sourceforge.net/softwaremap/trove\\_list.php?form\\_cat=6](http://sourceforge.net/softwaremap/trove_list.php?form_cat=6)
- (3) <http://www.thedirks.org/v4l2/>
- (4) <http://java.sun.com/j2se/1.3/ja/docs/ja/guide/jni/>
- (5) [http://www.hitachi.co.jp/Div/omika/prdcts/h-ip/ip5000\\_0.htm](http://www.hitachi.co.jp/Div/omika/prdcts/h-ip/ip5000_0.htm)
- (6) <http://www.interface.co.jp/catalog/prdc.asp?name=pci-5520>