ImageMagick

my recommendations on research & development tools.

高橋 克彦†

キーワード：画像変換，一括処理，CUI，フリーソフトウェア，オープンソース

1. ImageMagickとは

ImageMagick¹⁾とは、画像の生成，編集，合成，フィルタリングなどを行うソフトウェア群である。JPEG，JPEG-2000，BMP，GIF，PNM^{*1}，TIFF，PNG，PostScriptなどの100種類を超える多くの画像形式の読み書きに対応しており，相互に形式変換することも可能である。

ImageMagickは，Adobe Photoshop，Gimp，IrfanViewなどのGUIを備えたソフトウェアとは異なり，操作は基本的にコマンドラインで行う。そのため，軽快でバッチ処理などの分野で特に威力を発揮する。ImageMagickは，GPL互換のライセンスで公開されているフリーソフトウェアで，UnixやMicrosoft Windows，Mac OS X上で動作させることができる。多くのUnix環境で，バイナリーパッケージが用意されており，Windowsでも，開発元がインストーラを用意しているので，簡単に始めることができるだろう。もちろん，ソースコードからコンパイルすることもできる。

私の研究室では，本ソフトウェアを主として複数ファイルを対象とした形式変換や切出し，解像度変換などに使用している。

2. 対応する画像形式

対応している画像形式は，ImageMagickのバージョンやコンパイル時のオプションなどで変わってくる。使用環境におけるImageMagickの対応画像形式を調べるには，下記のコマンドを実行すればよい。

```
$ convert -list Format
Format Module Mode Description
-----
A* RAW      rw+ Raw alpha samples
ART* ART     r- - PFS: 1st Publisher
AVI* AVI     r- - Microsoft Audio/Visual Inte
AVS* AVS     rw+ AVS X image
B* RAW      rw+ Raw blue samples
```

*1 主にUnixで利用されている画像形式PPM，PGM，PBMの総称。

† 東京農工大学 工学府 電気電子工学専攻 高木研究室
"ImageMagick" by Katsuhiko Takahashi (Department of Electrical and Electronic Engineering, Graduate School of Engineering, Tokyo University of Agriculture and Technology, Tokyo)

```
BMP* BMP      rw- Microsoft Windows bitmap im
(省略)
YCbCr* YCbCr  rw+ Raw Y, Cb, and Cr sample
YCbCrA* YCbCr  rw+ Raw Y, Cb, Cr, and opacity
YUV* YUV      rw- CCIR 601 4:1:1 or 4:2:2
```

Mode欄のr，w，+は，それぞれ読み込み可能，書き込み可能，マルチイメージ可能を意味している。マルチイメージとは，複数のフレーム（ページ）を含むファイルのことである。

3. コマンドの使い方

ImageMagickのコマンド群には，animate，compare，composite，conjure，convert，display，identify，import，mogrify，montage，streamがあるが，ここでは，よく使うコマンドについて説明する。いずれのコマンドも-helpオプションを付けて実行することで，簡単な使用方法が表示されるので，manpageなどとともに参考にしていただきたい。

3.1 display: 画像の表示

```
display [options] input-file
```

display コマンドは画像の表示を行う。

```
display input.jpg
```

のように，表示したい画像を引数として起動する。

表示中は以下のキーボードアクセラレータが使用可能である。

[Space] 次の画像を表示

[BackSpace] 前の画像を表示

[Q] プログラム終了

[<] 縮小表示

[>] 拡大表示

[=] オリジナルサイズ表示

[/] 時計回りへ90°回転

[\] 反時計回りへ90°回転

[?] 画像の情報を表示

また，マウスの中ボタンを押して，カーソル位置を拡大して表示したり，カーソル位置のピクセル値を表示したりす

図1 displayコマンドを用いた画像の表示
(一部を拡大して表示しているところ)

図2 displayコマンドを用いたGUI画像処理

することもできる(図1)。

このコマンドは、簡単なGUIの画像処理を行うこともできる。画像を表示しているウィンドウ上でマウスの左ボタンを押すことで、メニューが表示される。そこで、TransformやEnhance, Effects, F/X, Image Edit以下の項目を選んで実行する(図2)。後述する一括処理を行う前に効果を確認するのに便利であろう。

3.2 animate: 複数画像のアニメーション表示

```
animate [options] input-file...
```

animateは、画像のアニメーション表示を行う。連番ファイルに記録された動画を順番に見ていったり、アニメーション表示させて確認したりするのに便利であろう。

例えば、画像ファイルinput-00.ppm, input-01.ppm, ...があるディレクトリで、

```
animate -delay 30 input-*.ppm
```

と実行すれば、画像を順番にアニメーション表示できる*2。ここでは、-delayオプションで、フレーム間の遅延を指定している。

なお、表示中は以下のキーボードアクセラレータが使用

*2 ここでの*は、シェルのワイルドカードである。

可能である。

- [Space] 画像を1枚ずつ順方向に表示。
- [BackSpace] 画像を1枚ずつ逆方向に表示。
- [<] フレームレートを上げる。
- [>] フレームレートを下げる。
- [Escape], [Q] プログラム終了。

3.3 identify: 画像情報の表示

```
identify [options] input-file...
```

identifyは、画像ファイルの形式や解像度、深度、圧縮方法などの情報を表示する。-verboseオプションとともに実行することで、より詳しい情報を調べることができる。

```
$ identify -verbose input.jpg
```

Image: input.jpg

Format: JPEG (Joint Photographic Experts Group JFIF format)

Class: DirectClass

Geometry: 3872x2592

Type: TrueColor

Endianness: Undefined

Colorspace: RGB

Channel depth:

Red: 8-bit

Green: 8-bit

Blue: 8-bit

(省略)

3.4 convert: 画像の変換

```
convert input-file [options] output-file
```

convertは、画像の形式変換や変形、エフェクトの適用を行う。

3.4.1 画像形式変換

画像形式の変換は、下記のように入出力のファイルを指定して実行する。画像形式は、ヘッダや拡張子から推定される。

```
convert input.jpg output.tiff
```

JPEG形式をTIFF形式へ変換

```
convert input.tiff +compress output.ppm
```

TIFF形式をPPM (バイナリー) 形式へ変換

```
convert input.tiff -compress none output.ppm
```

TIFF形式をPPM (ASCII) 形式へ変換

```
convert input.ppm -compress LZW \
```

-depth 16 output.tiff

PPM形式を16ビットTIFF (LZW 圧縮) 形式へ変換

3.4.2 画像サイズ変換

```
convert input-file -geometry w x h output-file
```

```
convert input-file -geometry w x h! output-file
```

```
convert input-file -geometry r% output-file
```

図3 -geometryオプションによる画像サイズの変換

-geometryは、画像サイズの変換を行うオプションである。 $w \times h$ のように！を付けずに実行すると、元の画像のアスペクト比を保ったままで、 $w \times h$ のサイズに収まるように変換される。一方、 $w \times h!$ のように！を付けて実行すると、 $w \times h$ のサイズに強制的に変換される(図3)。また、 $r\%$ のように、元画像を100%とした比率で指定することもできる。

```
convert input.ppm \
  -geometry 640x640 output.ppm
      アスペクト比を保ったままサイズ変換
convert input.ppm \
  -geometry 640x640! output.ppm
      強制的に640×640のサイズへ変換
convert input.ppm -geometry 50% output.ppm
      1/2のサイズへ変換
```

3.4.3 画像の切出し

```
convert input-file -crop wxh+x_offset+y_offset output-file
```

-cropは画像の一部分を切り抜くオプションである。引数の意味は、「 $(x_{\text{offset}}, y_{\text{offset}})$ を開始点とした、 $w \times h$ の範囲」を意味している(図4)。

```
convert input.ppm \
  -crop 320x240+20+40 output.ppm
      座標 (20, 40) から320×240の範囲を切出す
```

3.5 mogrify: 画像の一括変換

```
mogrify [options] input-file...
```

convertが、入力ファイルと出力ファイルを両方指定して画像処理を行っていたのに対し、mogrifyは、処理結果

図4 -cropオプションによる切出し

を入力ファイルへ上書きする。オプションで指定した画像処理を複数のファイルに対して適用する際に使用する。画像処理のオプションはconvertと同様のものが使用可能である。

```
mogrify -geometry 640x480! -gamma 0.8 *.jpg
      複数ファイルのサイズ変換とγ補正をして上書き
```

mogrifyは、標準で入力ファイルへ処理結果を上書きするが、下記のように-formatオプションを使って他形式の出力画像形式を指定すれば、上書きされない。出力ファイル名は、入力ファイル名の拡張子を変更したものになる。

```
mogrify -format jpg img1.ppm img2.ppm
      PPM形式をJPEG形式へ変換
      (img1.ppm→img1.jpg, img2.ppm→img2.jpg)
```

3.6 ファイル名の修飾子

ファイル名の前後に修飾子を付けることで、各コマンドでの画像の入出力の動作を変えることができる。

3.6.1 マルチイメージファイルで一部分のみ読み込む

下記のように、[]を用いることで、複数フレーム(ページ)からなるファイルの読み込みを一部のフレーム(ページ)に限定することができる^{*3}。

```
convert input.avi '[10]' output.jpg
      AVIファイルの10フレーム目をJPEG形式へ変換
convert input.pdf '[0,5-7]' output.tiff
      pdfファイルの0, 5~7ページをTIFF形式へ変換
```

3.6.2 画像形式の指定

入出力の画像形式は、ファイルのヘッダや拡張子から推定されるが、明示的に指定することもできる。以下のようにformat:をファイル名の前に付ける。

```
convert PPM:input TIFF:output
      PPM形式で読み込みTIFF形式へ変換
```

3.6.3 標準入出力

ファイル名の代わりに-を指定することで、画像を標準

*3 「[]」は、シェルによる展開を防ぐために付けている。

入力から読んだり、標準出力へ書き出したりすることができる。

```
cat input.ppm | \
convert - -geometry 320x240 PPM:- | \
display -
```

この例では、シェルのパイプを用いて、コマンドをつないでいる。convertは、標準入力からファイルを読み込み、サイズを変換して、PPM形式で標準出力へ画像を書出す。それをdisplayは、標準入力から読んで表示する。

4. 応 用 例

ここでは、いくつかの応用例を示す。説明していないコマンドも出てくるが、用法についてはmanpageなどを参照していただきたい。

4.1 ヘッダのない画像ファイルの読み込み

ヘッダを持たないRaw形式の画像では、読み込みの際、サイズや深度の情報を与える必要がある。これらの情報は、-sizeオプションと-depthオプションで与える。下記の例では、640×480のRGB各色8ビットのRaw画像を読み込んで変換している。

```
convert -size 640x480 -depth 8 \
RGB:input.raw output.ppm
```

4.2 サムネイル画像の一括作成

Webページ向けにサムネイル画像を作成する例を紹介する。thumbというディレクトリを作成して、その中に最大で64×64のサイズに収まるようなサムネイル画像を作成する。また、圧縮する際の画像品質も指定している。

```
mkdir thumb
for img in *.jpg
do
    convert -geometry 64x64 \
        -quality 60 $img thumb/$img
done
```

4.3 連番ファイルへの書き出し

AVIファイルに記録された動画像を読み込み、1フレームずつファイルに書き出す例を紹介する。

```
convert input.avi \
+adjoin frame-%03d.ppm
```

この例では、C言語のprintf関数と同様の形式で、出力ファイル名を指定している。そのため、先頭フレームから順に、frame-000.ppm, frame-001.ppm, ...というファイル名で出力される。

4.4 Animated GIFの作成

Animated GIF形式の作成方法を紹介する。多くのWebブラウザが対応しているので、Webページ作成の際に役立

つであろう。

```
convert -loop 0 -delay 6 input.avi output.gif
          AVIファイルをAnimated GIFへ変換
convert -loop 0 -delay 12 \
    00.ppm 01.ppm 02.ppm 03.ppm 04.ppm \
    -adjoin output.gif
          複数のPPMファイルからAnimated GIFへ変換
```

この例では、-loop、-delayオプションで、それぞれループ回数(無限ループの場合は0)、遅延時間を指定している。

4.5 PSNR の計算

2枚の画像を比較してPSNR (Peak Signal-to-Noise Ratio) を計算するには、下記のようにcompareコマンドを利用する。

```
compare -metric PSNR \
img1.ppm img2.ppm null:
```

4.6 スクリーンキャプチャ

スクリーン表示を画像ファイルとして保存するには、importコマンドを使用する。コマンドを実行して、キャプチャしたいウィンドウ上でマウスをクリックすればよい。スクリーン全体を取得するには、-window rootを付ける。

```
import window.png
          マウスでクリックしたウィンドウのみ取得
import -window root screen.png
          スクリーン全体を取得
```

5. む す び

ImageMagickは、非常に多くの機能を持ったソフトウェアである。今回は、そのうちのよく使用する機能や、コマンドラインでの使用が効果的なものに絞って説明した。より詳しい説明は、開発元サイト¹⁾のドキュメントやmanpageなどを見ていただきたい。

今回は紹介できなかったが、ImageMagickは、ライブラリとしても公開されている。このライブラリは、C 言語、C++、Perl他、多くの言語から利用することができる。これらを用いることで、多形式に対応した画像入出力処理やさまざまなアルゴリズムを容易に実装できるであろう。

(2007年10月3日受付)

〔文 献〕

1) ImageMagick開発元サイト, <http://www.imagemagick.org>

高橋 克彦 2006年、東京農工大学工学部電気電子工学科卒業。現在、同大学院工学府電気電子工学専攻修士課程に在学中。主として、3次元画像処理の研究に従事。