Pink
A User Manual

Michel Couprie, László Marak & Hugues Talbot

Version 0.1 as of April 15, 2013

Documentation is like sex:

when it is good, it is very, very good;

and when it is bad, it is better than nothing.[1].

À nos pauvres étudiants qui ont été forcés de souffrir pendant des heures sur ce logiciel.

---

[1]Attributed to Dick H. Brandon, pioneer in computer law.

Contents

Chapter 1

Introduction

PINK is an image processing library developed at ESIEE Paris for research and teaching purposes. It contains implementations of over 200 algorithms for image segmentation and filtering. Most of the operators come from mathematical morphology, but it contains operators from different fields. Pink is free software licensed under the CeCILL license.

We are interested in the continuous development of Pink. It has already been proven useful in many applications and we are constantly looking for new ones.

This manual aims at referencing most functions of PINK, hopefully in a didactic manner.

History

Pink is Not Khoros

Related software

These days, image processing is a technology and something such as PINK is best used as a component among others. For optimal use, the following packages should be installed:

- imview

- Python (version 2.7 preferred) ; with the following packages:

  - numpy
  - scipy
  - matplotlib
  - python-vtk
  - python-image (PIL)
  - python-image-tk

- Doxygen

- ActiveTcl 8.3

- VTK

- MPlayer

- Gnuplot

Other software may prove useful:

- OpenCV

- ITK

Contributors

PINK is the result of many thousands of hours of work, and includes contributions from this (non-exhaustive) list of people

Code under the main CeCILL license:

- Michel Couprie : main author, initial design

- László Marak (ujoimro) : library, port to Python, continuous maximum flows, Total-Variation denoising, Python front-end, native Microsoft Windows port.

- Laurent Najman : localextrema, saliency

- Hugues Talbot : fmm, fast morphological operators, region growing; this documentation.

- Jean Cousty : redt 3d (reverse euclidean distance transform - algo de D. Coeurjolly), watershedthin, opérateurs sur les graphes d'arêtes (GA), minimum cost forest (MSF), waterfall, recalagerigide translateplane

- Xavier Daragon: dist, distc (Quadratic Euclidian Distance in 3D)

- André Vital Saude: radialopening, divers scripts tcl, hma

- Nicolas Combaret: toposhrinkgray, ptselectgray

- John Chaussard: lballincl, cropondisk, shrinkondisk

- Christophe Doublier: zoomint

- Hildegard Koehler: lintophat

- Cédric Allène: gettree, histolisse, labeltree, nbcomp, pgm2vtk, seuilauto

- Gu Jun: maxdiameter

- Sébastien Couprie: mcsplines.c

- Rita Zrour: medialaxis (exact Euclidean medial axis - algorithm of Rémy Thiel), dist, distc (Quadratic exact Euclidean distance - algorithm of Saito-Toriwaki, in 2D)

- Laurent Mercier: gestion d'un masque dans delaunay

- Benjamin Raynal: parallel 3D thinning

- Nivando Bezerra: parallel grayscale thinning

Code under different free software licenses:

- David Coeurjolly: lvoronoilabelling.c

- Dario Bressanini: mcpowell.c

- Andrew W. Fitzgibbon: lbresen.c

- Lilian Buzer: lbdigitalline.cxx

Chapter 2

Tutorial

This chapter explains how to start PINK and how to use it.

2.1 Starting PINK and related software

2.2 Examples

We will show a lot of Python code:

2.2.1 Code highlighting

Testing Listings:

```python
#!/usr/bin/env python
#

print("Hello World of Pink/Python\n")
```

Testing fancyvrb

Program 1: hello in Python

```python
1  #!/usr/bin/env python
2  #
3
4  print("Hello World of Pink/Python\n")
```

2.2.2 Dynamic content

Some code and its output: The following program produces the plot from Fig. 2.1.

Program 2: A plotting example

```python
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  fig = plt.figure()
5  ax1 = fig.add_subplot(111)
6  t = np.arange(0.01, 10.0, 0.01)
7  s1 = np.exp(t)
8  ax1.plot(t, s1, 'b—')
9  ax1.set_xlabel('time (s)')
10 ax1.set_ylabel('exp')
```
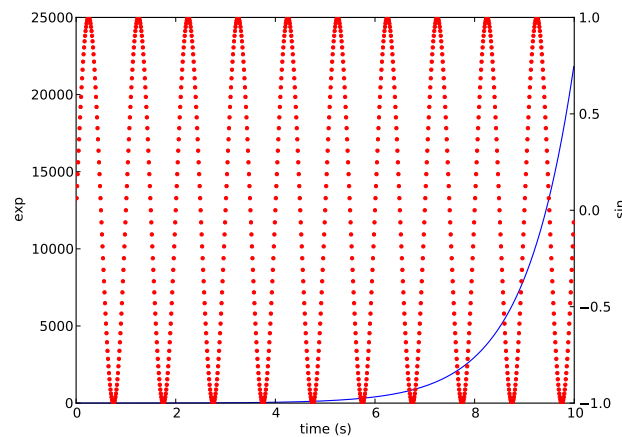
Figure 2.1: The functions sin and exp together from program 2.

```
11
12 ax2 = ax1.twinx()
13 s2 = np.sin(2*np.pi*t)
14 ax2.plot(t, s2, 'r.')
15 ax2.set_ylabel('sin')
16 #plt.show()
```

2.2.3 Image processing

An example of processing?

2.2.4 Binary image processing

2.2.5 Grey-level segmentation

2.2.6 Discrete geometry

2.2.7 3D image processing

2.3 PINK and other image-related software
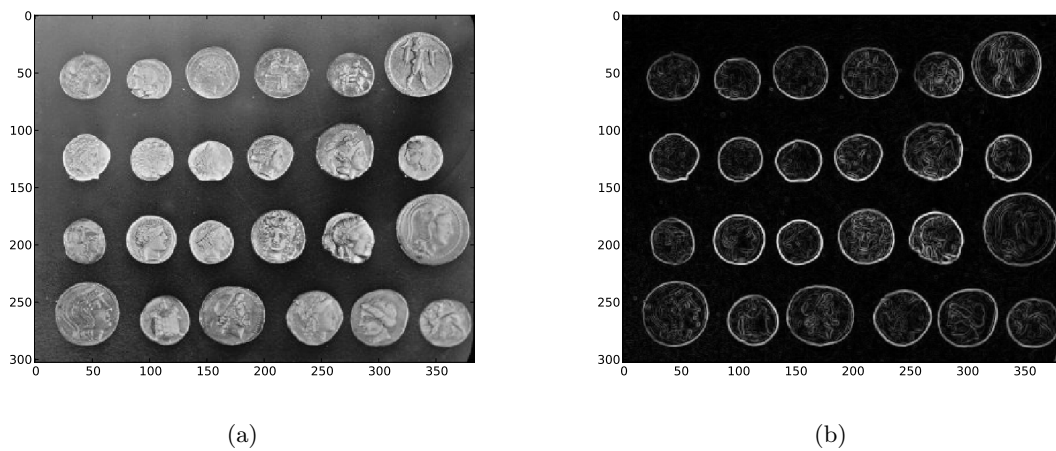
2.3.1 PINK and numpy

2.3.2 PINK and scikit

2.3.3 PINK and VTK

2.3.4 PINK and ITK

2.4 Exercises

(a)

(b)

Program 3: An image processing example

```python
from skimage import data, filter, io

# not hard !!
def sampleproc(image):
    edges = filter.sobel(image)
    return edges

## saving procedure
def saveimages(path1, path2):
    inputimg = data.coins()
    outputimg = sampleproc(inputimg)
```

Figure 2.2: Program 3 and its result.

Chapter 3

Complementary python tools: numpy, scipy, scikit-image, etc

This chapters describes working with complementary python tools, such as numpy, scipy, the scikits, etc.

In recent years, python has become a credible replacement for Matlab, thanks to a huge community effort in porting basic numerical facilities to python. Other efforts such as those of Enthought, a company that provides a well-designed distribution of python oriented towards parallel numerical tools, have made python a performance tool suitable for many enterprise-level applications.

The aim of this section is to show how these tools complement PINK.

3.1   Converting from/to numpy

The *lingua franca* of numerical computing under python is numpy. PINK provides means to swap back and forth between its internal format and numpy arrays.

Program 4: Numpy conversion

```
1 #import pink
2 #import numpy as np
3
4 # creation of some 3D data
5 myarray = np.array(np.arange(0,5*5*5)).reshape(5,5,5)
6 myarray
7 # convert array -> pink
8 myimage = pink.numpy2pink(myarray)
9 # convert pink -> array
10 newarray = pink.pink2numpy(myimage)
11 newarray
12 # variance calculation
13 newarray.var(axis=0)
```

The plan for future versions of PINK is to make this conversion automatic and transparent, so that for the Python user, PINK image data is always a numpy array.

3.2   Basic numpy usage

Appendix A

Compiling and installing PINK

Not as difficult as it may seem.

LICENSE

This document "PINK, A USER MANUAL"by Michel Couprie, Laszlo Marak and Hugues Talbot is licensed under a Creative Commons Attribution 3.0 Unported License

This means, you are free:

- to Share — to copy, distribute and transmit the work

- to Remix — to adapt the work

- to make commercial use of the work

Under the following conditions:

- Attribution — You must attribute PINK A USER MANUAL to Michel Couprie, Laszlo Marak and Hugues Talbot (with link: `http://pinkhq.com/`).

With the understanding that:

- Waiver — Any of the above conditions can be waived if you get permission from the copyright holder.

- Public Domain — Where the work or any of its elements is in the public domain under applicable law, that status is in no way affected by the license.

- Other Rights — In no way are any of the following rights affected by the license:

    - Your fair dealing or fair use rights, or other applicable copyright exceptions and limitations;
    - The author's moral rights;
    - Rights other persons may have either in the work itself or in how the work is used, such as publicity or privacy rights.

Notice — For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page: `http://creativecommons.org/licenses/by/3.0/`

List of Figures

List of Tables