Suceed Hackathon

Open source OCR & line detection

Jesús L. Domínguez Muriel

Open Source OCR status

- Tesseract – C++
- Cuneiform – released as freeware by Cognitive Technologies. Large code base, not easy to find, in russian ☹
- GOCR
- OCRAD C++ http://ftp.gnu.org/gnu/ocrad/. Very small,
- OCROpus

Ocrad

- GNU OCR
- http://ftp.gnu.org/gnu/ocrad/
- C++
- Very simple line recognizer: blob detection -> enlarging -> joining
- Textpage.cc, 547 lines, readable and usable

OCR Opus

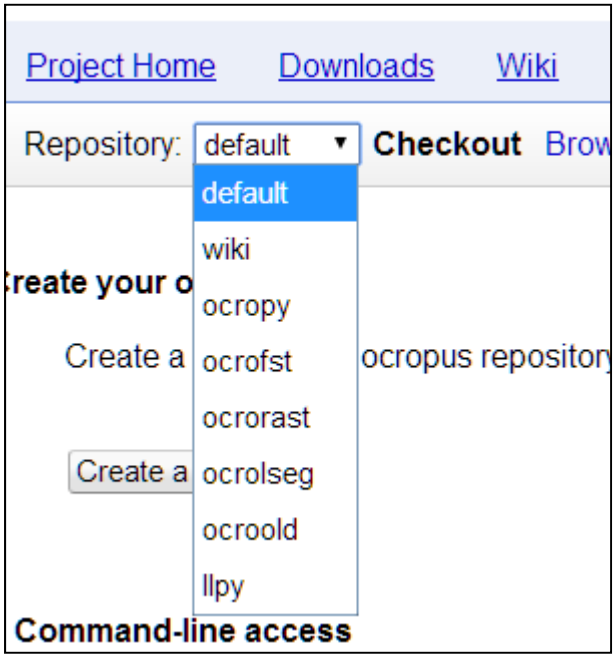
- https://code.google.com/p/ocropus/
- Not downloadable today with Chrome (OpenSSL Heartbleed vuln.?)
- Internet Explorer can save your day
  - » Surprising, yes!

OCR Opus

- OCRopus™ is an OCR system written in Python, NumPy, and SciPy focusing on the use of large scale machine learning for addressing problems in document analysis.
- Current version 0.7, not very active (2012)
- Great introduction: http://nbviewer.ipython.org/url/ocropy.ocropus.googlecode.com/hg/Notebooks/ocropus-steps.ipynb

Code

- Several repositories on code.google.com
- Default repository includes only test files
- Main code: ocropy
- Old segmentation code in ocrolseg (C++ and Python)
- Several methods



Python libraries

- Heavy use of Scientific libraries: SciPy, NumPy.
- Not easy to run under Windows
- Standard Python package manager pip reports errors downloading SciPy
- WinPython distribution to the rescue!

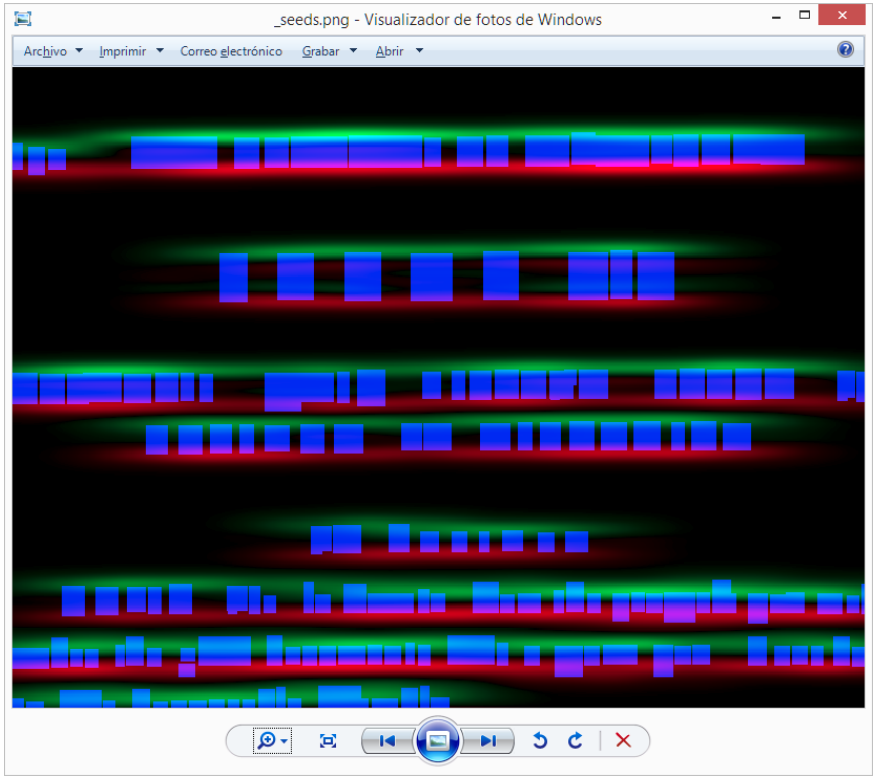
OCR Opus line detection

- ocropus-gapageseq
- Difficult to reuse outside Python due to use of a huge Python image analys and extraction library, SciPy ndimage
- Example: box map (rectangles containing non sur characters): 7 lines

```
objects = binary_objects(binary)
bysize = sorted(objects,key=sl.area)
boxmap = zeros(binary.shape,dtype)
for loop
```

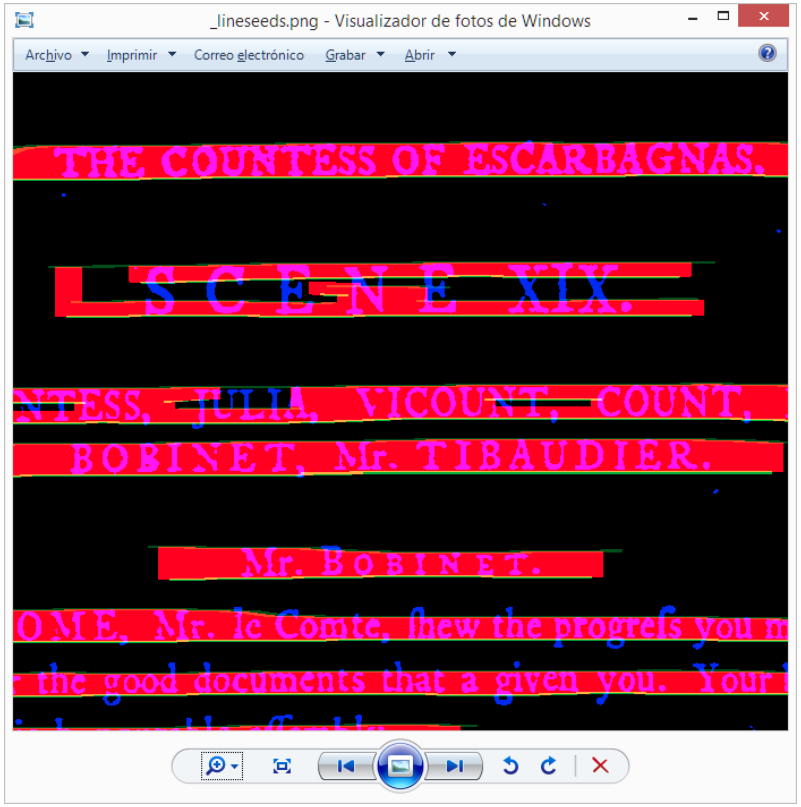
OCR Opus line detection

- Estimate scale of characters, compute boxmap
- Use gaussian filter to detect line top and bottoms (gradient map + adaptive thresholding)

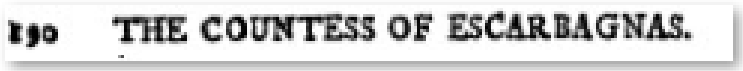


OCR Opus line detection

- Erosion and dilation to further refine top and bottom
- Then mixes the lines with the blocks to identify lines
- Good but no perfect



Good but no perfect



010001.bin.png

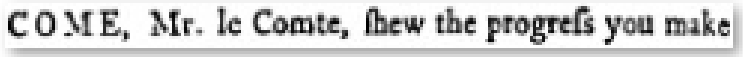
010002.bin.png

010003.bin.png

010004.bin.png

010005.bin.png

010006.bin.png



Final remarks

- Great to meet you all
- We need an algorithm Wikipedia – a list of real life implementations of useful algorithms
- I like Python