Image Signatures

Image Signatures are numerical descriptors of image content

- They describe image properties such as
  - Edges, Shapes, Textures, Regions
- We use AI machines to interpret the image signatures to judge image content

- Image Signatures are the basis for our lab’s research on Automated Image classification and scoring of High-Throughput High-Content Image screens
Computing Image Signatures

- There already exist state-of-the-art Computer Vision algorithms for measuring various image properties:
  - Edges, Shapes, Textures, Regions
- We’ve collected MATLAB implementations of a dozen such algorithms
- Use-Case: run our algorithms on hundreds of images to feed our AI machines.
  - Lots of Images per Collection (hundred)
  - Lots of Image Collections (dozen)
  - Many algorithms, computationally intensive, and arranged in a complex workflow
- Goal: We need the signatures to be “easily” and “quickly” (tools generate results quickly)
  - Easily: It is easy to teach biologists how to compute signatures
  - Quickly: The tools are computationally efficient
  - it goes without saying: Validation!
- How are we accomplishing this? Using OME Technology of course!
Signature Chain Schematic

- Original Files
- Image Import
  - Files
  - Pixels
  - Dimensions
  - Metadata
- Fourier Transform
  - Pixel Plane
  - Frequency Space
  - [m n], image size
- Wavelet Transform
  - Pixel Plane
  - Wavelet Coefficients
  - [m n], image size
- Chebyshev Transform
  - Pixel Plane
  - Order
  - Coefficient Matrix
  - [m n], image size
- Comb-4-Moments Histogram
  - Pixel Plane
  - Histogram vector
  - 48 values
- Zernike Moments
  - Pixel Plane
  - Order
  - Zernike Moments
  - 72 values
- Chebyshev-Fourier Statistics
  - Pixel Plane
  - Coefficient Histogram
  - 30 values
- Chebyshev Statistics
  - Pixel Plane
  - Order
  - Coefficient Histogram
  - 30 values
- Harellick Features
  - Pixel Plane
  - Text. direction
  - 24 values
- Radon Statistics
  - Pixel Plane
  - Transform Histograms
  - 12 values
- Multi-scale Histograms
  - Pixel Plane
  - Histogram vector
  - 16 values
- Tamura Statistics
  - Pixel Plane
  - Texture Statistics
  - 6 values
- Gabor Statistics
  - Pixel Plane
  - Histogram, 9 values
  - Ratio score, 6 values
- Gradient
  - Pixel Plane
  - Gradient
  - [m n], image size
- Global Threshold
  - Pixel Plane
  - Threshold
  - Binary Mask
- Edge Statistics
  - Gradient
  - 8 scalar values
- Feature Statistics
  - Binary Mask
  - 16 scalar values
Data Modeling

- Our use-case requires both strong semantic typing and generality; e.g.:
  - Outputs from Fourier Transform and Wavelet Transform must be stored in OMEIS, and should be treated (sometimes) as Pixels. However these outputs must not be modeled as Pixels ST.
  - This has motivated our home-brewed method of inheritance: Type-Casters and Parental Types
Parental Types

- Fourier Transform and Wavelet Transform ST definitions:

Parental Types are a convention supported by the MATLAB Handler and Typecaster Modules (src/perl2/OME/Analysis/Modules/PPM)
Typecaster Modules

- Typecaster Modules are defined in XML by specifying Formal Inputs and Outputs. There’s a general Perl package that serves as a general Typecaster module implementation.

src/perl2/OME/Analysis/Modules/PPM/Typecast2Pixels.pm
Although image signatures are typically used to describe the whole image’s content, it is equally valid to compute signatures just on an image region.

Our lab’s use-cases require computing signatures per-channel and per image tile. Hence we have developed modules (ROI constructors) in Perl for programmatically fragmenting the images. The signature chain is data-modeled to compute signatures per ROI.

Let us agree on a common ROI data-model that is usable by manual ROI tools (e.g. Java ROI UI tool), automated ROI modules (e.g. ROI Constructors), and OME Analysis Chains that need to operate on them (e.g. Signature Analysis Chain).
ROI Data model

• Whole image or ROI?

• Our ROI constructors take an image’s Pixels and generate Feature Granularity attributes of ST DerivedPixels.

• We provide three ROI Constructor modules (Whole Image ROI, 2D Image ROI, 2D Tiled Image ROI). These modules are defined in Perl (src/perl2/OME/Analysis/Modules/ROI) and programmatically generate ROIs in various ways.

• ROI constructor modules generate Features called e.g. Whole Image ROI, 2D Image Tile 0 0, 2D Image Tile 0 1, 2D Image Tile 1 1 etc.

• DerivedPixels (F) ST has Parent Pixels reference & extents.

src/xml/OME/Analysis/ROI/DerivedPixels.ome
Discuss Signature Data-Model
MATLAB + OME

more info:


http://users.openmicroscopy.org.uk/~tmacur1/Macura_OME_MatlabHandler.pdf

contact person:

Tom Macura tmacura@nih.gov
http://users.openmicroscopy.org.uk/~tmacur1/tomasz_goes_to_washington.jpg
MATLAB

• MATLAB is a high-level, interactive programming environment for numerical computation. With more than 1000 built-in mathematical, statistical, and engineering functions, including statistical, image analysis, and machine learning tools, MATLAB is especially useful for quantitative bioinformatics. Our lab uses MATLAB as:

  – an interactive environment for data analysis and visualization leading to algorithm development

  – a programming language that allows us to implement numerical analysis algorithms much faster than with traditional programming languages such as C, C++, or Fortran.
OME/MATLAB coupling

We designed the OME/MATLAB coupling to support both MATLAB usage methods:

- Developing algorithms in MATLAB using OME data (omeis-http)
  http://www.openmicroscopy.org.uk/api/omeis/matlab-mex.html

- Using MATLAB for parallel execution of high-throughput workflows defined in OME (MATLAB Handler)
  http://www.openmicroscopy.org.uk/api/xml/AML/matlab.html

- Recipe for using OME to compute image signatures
Optional OME-MATLAB Setup

(All verbose information logged in /var/tmp/OME/install/MatlabTask.log)

Testing MATLAB configuration
\_ Checking if path to MATLAB .m files is stale [FAILURE].
\_ Path was stale. Reset /home/tmacur1/OME-Hacking/OME/src/matlab to /home/tmacur1/Fresh/OME/src/matlab
\_ Checking if user apache is licensed to run /usr/local/bin/matlab [SUCCESS].

MATLAB Perl API configuration:
\_ Install MATLAB Perl API?: yes
  MATLAB User: apache
  MATLAB Path: /usr/bin/MATLAB7SP2
  MATLAB Exec: /usr/local/bin/matlab
  MATLAB Exec Flags: -nodisplay -nojvm
  Config MATLAB for dev?: yes
  MATLAB .m files Path: /home/tmacur1/Fresh/OME/src/matlab

Are these values correct? [y/n]: y
\_ Double checking if user apache is licensed to run /usr/local/bin/matlab [SUCCESS].
\_ Gathering information about your MATLAB installation

Matlab Vers: 7.0.4.352
Matlab Arch: glnx86
Matlab Root: /usr/bin/MATLAB7SP2
Include: -I/usr/bin/MATLAB7SP2/extern/include
Lib: -L/usr/bin/MATLAB7SP2/bin/glnx86 -lmx -leng -lut -lmat -licuda -licuil8n -licuuc -lustdio -lz

Installing MATLAB Perl API
\_ Configuring [SUCCESS].
\_ Compiling [SUCCESS].
\_ Testing [SKIPPING].
\_ Installing [SUCCESS].

Installing MEX Interface to omeis-http
\_ Configuring omeis-http with MATLAB bindings [SUCCESS].
\_ Compiling omeis-http with MATLAB bindings [SUCCESS].

Installing OME MATLAB .m files
\_ Compiling MEX files [SUCCESS].
\_ Installing [SUCCESS].