

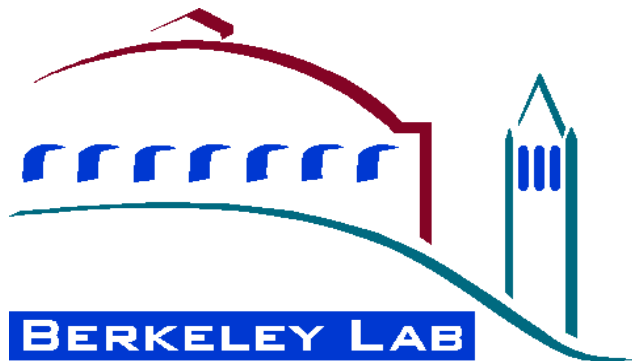
# History and Future of Computerized Data Acquisition: Application to Scanning Microscopy

D. Frank Ogletree, Ed S. Barnard

Imaging Facility

Molecular Foundry, Materials Sciences Division

Lawrence Berkeley National Lab



# A Short History of Computerized Experiments

## ◆ relatively "recent"

- only 30 years...
- STM developed at IBM research was ...analog...in early 80's

## ◆ mid-80's

- Artisanal or proprietary, limited hardware, almost no software tools, graphics/visualization, STM/AFM first computerized

## ◆ mid-90's

- crude SEM software, first TEM software without detector integration, CCD detectors for TEM and Spectroscopy...

## ◆ mid-2000's

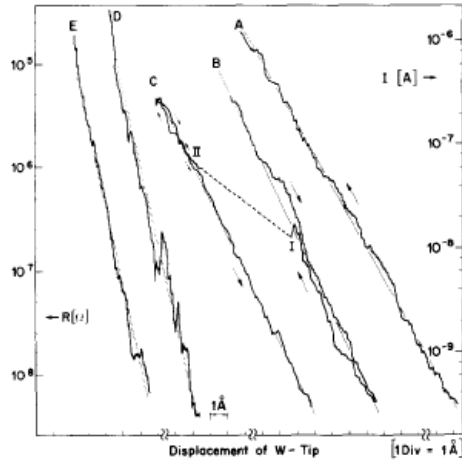
- internet, much better computers, operating systems, software environments, computer "literacy"

## ◆ mid-2010s,

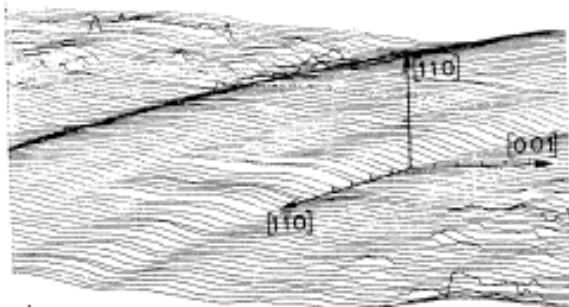
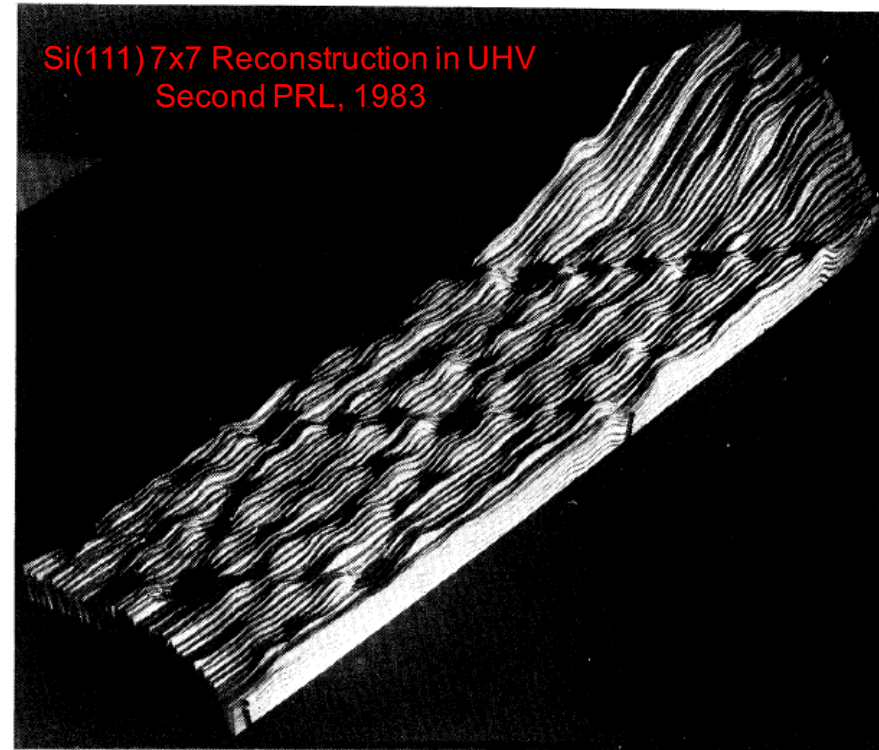
- high performance computing, fast networks, cheap storage, big data, theory/simulation much faster and more capable...

# Invention of the STM, 1981

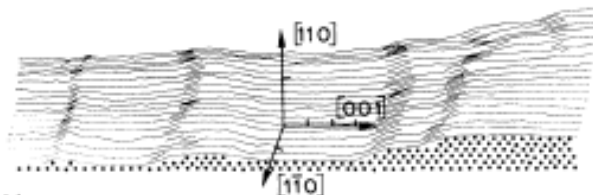
Gerd Binnig & Heine Rohrer, IBM R ushlikon



Vacuum tunneling between W tip and Pt foil, First APL, Binnig & Rohrer Jan 1982 (results from March 81)

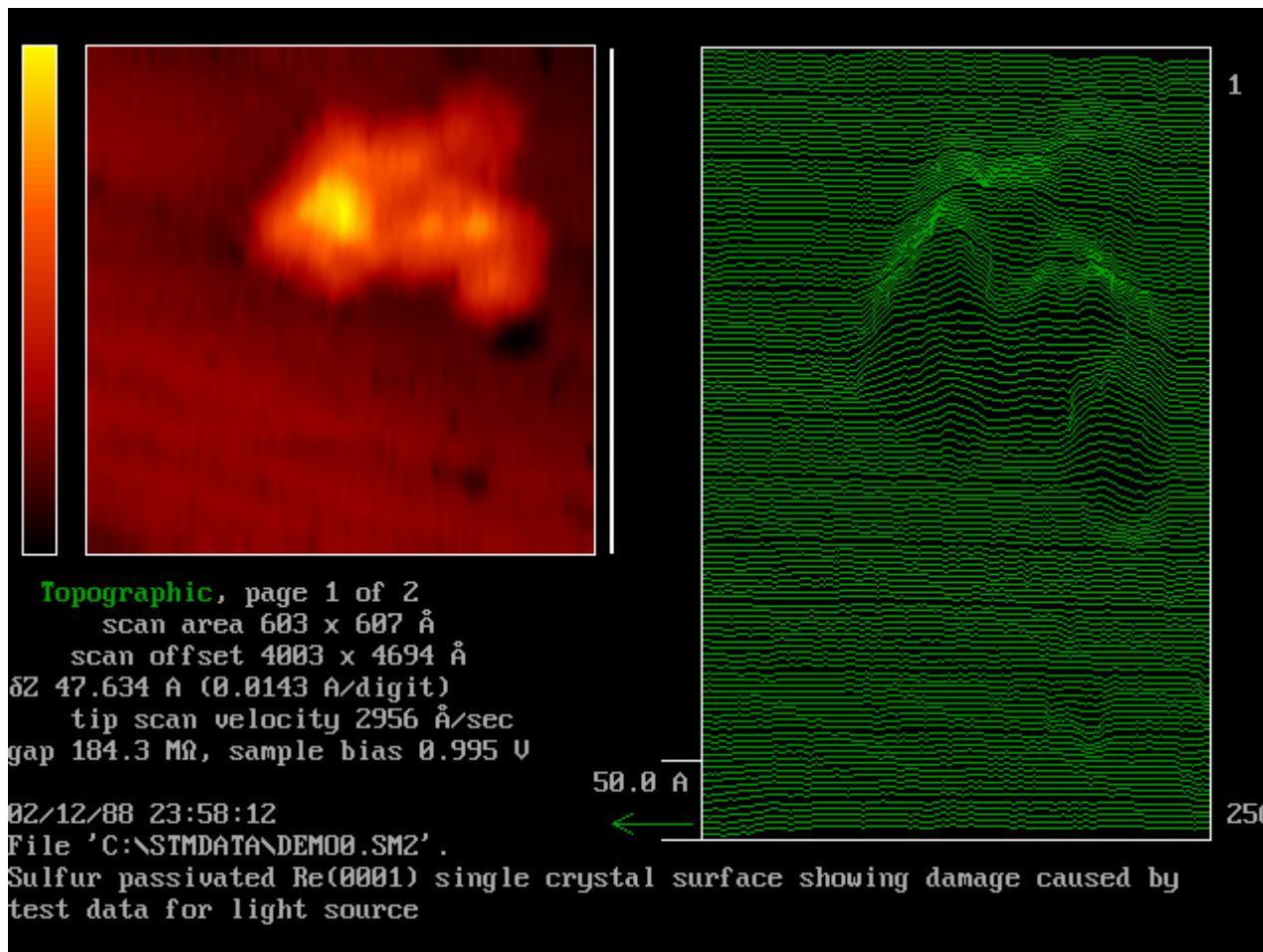


a)



b)

Atomic Steps on Au(110) in UHV First PRL, July 1982

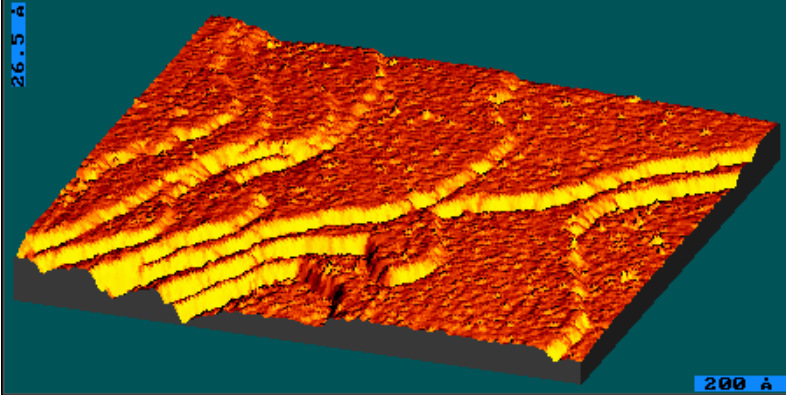


Screen capture of first STM program developed at LBL in 1987

Fortran on DEC LSI-11 minicomputer,  
 5 MB disk, 64 kB RAM, \$6,000 display system, 640x480 pixels

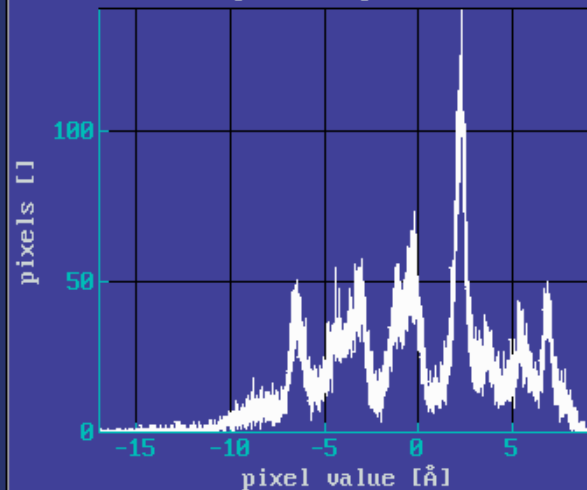
STiMage 3.13d Feb 28  
F1 main menu

DEMO05/1 Topographic Data

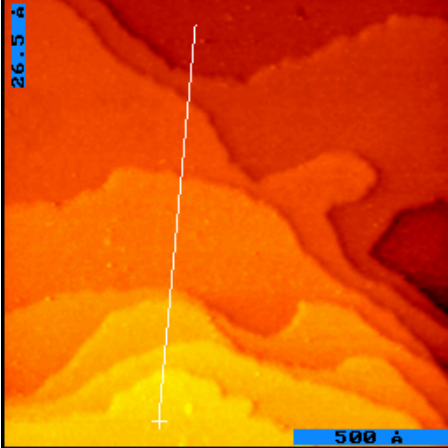


DEMO05/1 Topographic Data

Image Histogram



DEMO05/1 Topographic Data



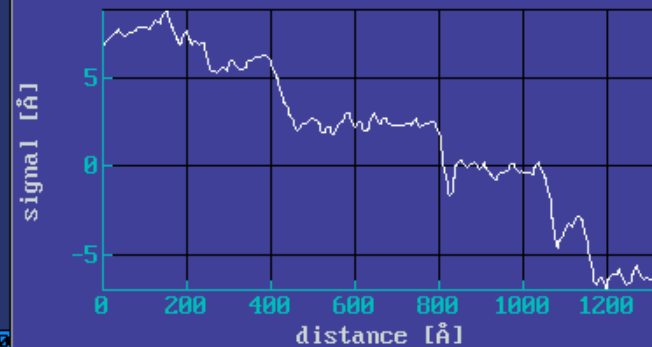
Cursor

x= -363 Å  
y= -370 Å  
z= -0.55 Å  
(63,64)  
z= 467  
δd 1334 Å  
δz -9.60 Å  
θ 84.9

Section  
+ label  
unlabel

DEMO05/1 Topographic Data

Cross Section (146,15) to (167,241)



DEMO14/1 F

DEMO01/1 T

DEMO02/1 C

DEMO04/1 T

DEMO11/0

STM program in 1993, C and Assembler on  
Compaq 80386 (\$19 k), 0.02 GHZ 1 MB RAM  
32 bit CPU, SVGA display, extended DOS

# Scanning Microscopy

- ◆ **Scanning Probe STM/AFM**
  - I-V, F-z, electrochemical, dissipation, acoustics, friction, piezo-responsive...
- ◆ **Confocal/Near Field Optical**
  - hyperspectral Raman, PL, PLE, lifetime, pump-probe, transient absorption, polarization, epifluorescence.....
- ◆ **Analytic SEM**
  - Cathodoluminescence, Quantitative current imaging/EBIC, Reflection EELS, Auger Spectroscopy, XRF/EDS/WDS, EBDC...
- ◆ **Analytic STEM**
  - EELS, XRF, CBED, BF/DF, SE, HAADF...
- ◆ **X-ray synchrotron methods**
  - STXM, SFXM...

## Data to Damage Ratio!

### ◆ SEM

- heating, radiation damage, contamination, charging (image and electronic properties)...

### ◆ STEM

- SEM modes plus lattice damage/atom displacement, ice radiolysis...

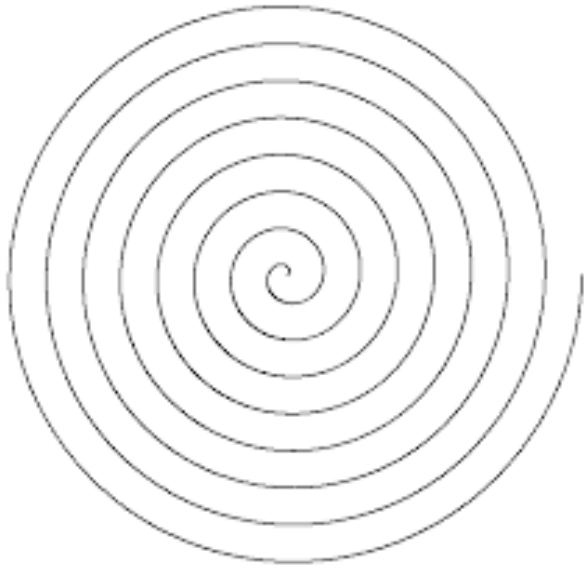
### ◆ STM/AFM

- tip change/wear, sample wear/contamination, tip-induced dynamic processes, vibrational excitations causing chemistry, diffusion...

### ◆ Optical

- thermal damage, melting/ablation, flurophore bleaching...

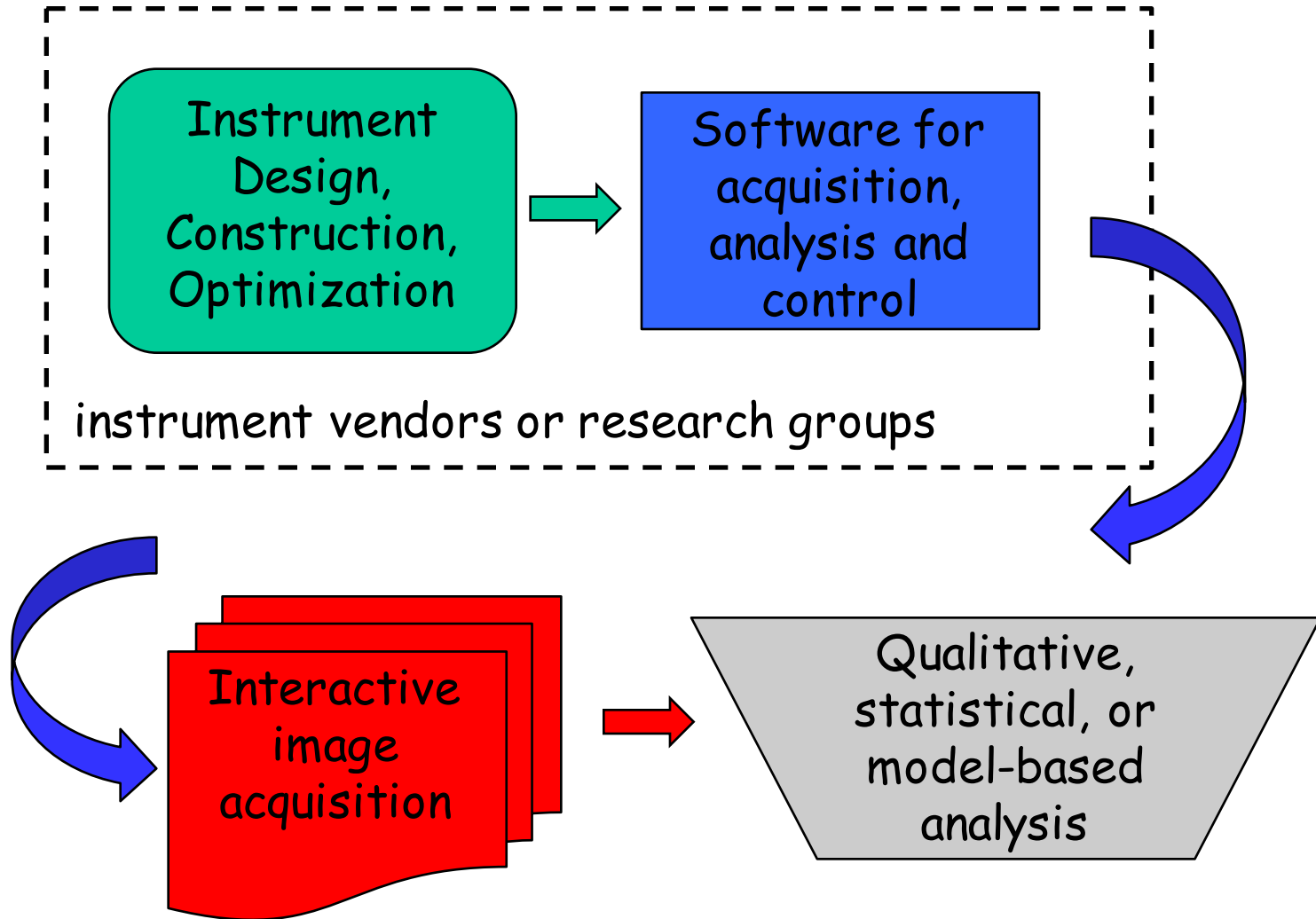
# Smarter Acquisition - Front End



- ◆ experiments not just images
- ◆ fast images - slow spectra
  - scan region once per spectral point
  - feature tracking during acquisition
  - depends on relative speed of instruments
- ◆ "adaptive" acquisition
  - automatic object finding, detail where its needed
  - low SNR image to find regions for hyperspectral mapping
  - SNR threshold not fixed time for spectra
- ◆ spiral scanning - Paul Ashby
  - edge detection/following



# Imaging Instrument Paradigm



# vendor instrument software

- ◆ often the “weak link”,
  - less capable than hardware
  - lags behind in software engineering, exponential growth in computing power
- ◆ SEM modify data before digitization/storage
  - no quantitative data, “contrast and brightness”, “channel mixing”, limited data channels
  - poor or no drift correction, no concept of spectroscopy, low dose imaging, copy “analog” video burn edge/corners
  - very minimal data visualization, pay extra for contrast...
- ◆ SPM
  - generally more powerful software but proprietary formats, can be unstable/crashes
  - limited scripting/programming (zB Asylum Igor, Nanonis Lab View)
- ◆ Optical microscopy
  - software mostly for bio imaging applications, sophisticated turn-key instruments, or build it yourself

## ◆ researcher developed solutions

- artisanal, strong integration science/function
- re-inventing the wheel, undocumented or oral tradition, user hostile...

## ◆ vendor software

- instruments with large customer/application base and competitive markets can have decent software for typical applications
- Often full power of hardware is “locked out”, unintended consequence or captive markets...
- scientific “niche” markets stuck with long software redesign cycles, “locked in” to bad/proprietary choices...

## ◆ Commercial software environments

- NI/Labview, Matlab...

## ◆ what is to be done?

# One Hardware/Software Challenge

## Cathodoluminescence

### ◆ SEM

- beam current/energy/focus
  - » SmartSEM GUI (computer #1), serial interface
  - » TTL beam blanker
- scanning/image acquisition
  - » external analog scan control inputs
- electron detectors
  - » analog and/or pulse count
  - » “classical” SEM single data stream

### ◆ extra acquisition/control

- » RHK SPMpro scanning, counter, multichannel data (computer #2)
- » Labview CCD, spectrometer, heater, (computer #2)
- » SRS electronics modules
- » Andor, Acton, Attocube, Camera, etc vendor software

### ◆ Optical Components

- collection mirror – attocube nano-translators
  - » TTL inputs (old)
  - » closed loop USB-DLL
- Acton grating spectrometer
  - » USB text commands
- Andor spectroscopic CCD
  - » USB-DLL
- Acton OMA V IR diode array
  - » USB-DLL
- optical point detectors
  - » PMTs, APDs, pulse train
  - » IR photodiodes, analog
- CMOS imaging camera

### ◆ Sample

- thermocouples, heaters, cryostat, Lakeshore controller
  - » GPIB, voltage programmed

# "ScopeFoundry" for Experiments

## ◆ Emerging platform for Experiments

- Developed by Ed Barnard (last talk) for confocal spectro-microscopy experiments
- Extended to fast experiments/acquisition on SEM/CL/Auger , NCEM
- Separate processes for instrument control, user GUI, data handling

## ◆ Include real instrument response functions ?

## ◆ Couple to HPC/Bigger data ?

- ORNL Beams??

## ◆ Include (real time) simulations of probe-sample interactions ??

- Physics mostly known, tools for calculation of different aspects mostly exist, rarely used (activation barrier, learning curve...)

# "ScopeFoundry" EcoSystem

## ◆ Scientific Python

- Anaconda for Mac, Windows, Ubuntu, almost pain-free setup
- Rapidly expanding open source toolset, connect to good numeric libraries, Device independent graphics Qt-Pyside
- Debug on the fly during experiments (Eclipse editor)

## ◆ Instrument control

- Support most common and obscure instrument interfaces
- call DLL drivers, Serial (GBIP, USB, RS-232, etc)

## ◆ Hardware

- National Instruments (DAQmx-Python)
- NI PXI-hosted FPGA fast decision making (C DLL-Python)
- Fast data transfer PXIe

## ◆ Data

- HDF5 (Python library) images/metadata/experiments

# Open Source Success ?

## ◆ How can viable software communities be created ?

- many good intentions, efforts and extinct projects, standards, environments..

## ◆ Example of ImageJ

- open source, multi-platform, extensible
- many 100s of contributors, many 1000s of users
- core of dedicated developers/coordinators, supported to some extent by NIH...
- “Quantum Espresso”, “NanoHub”, other academic projects....

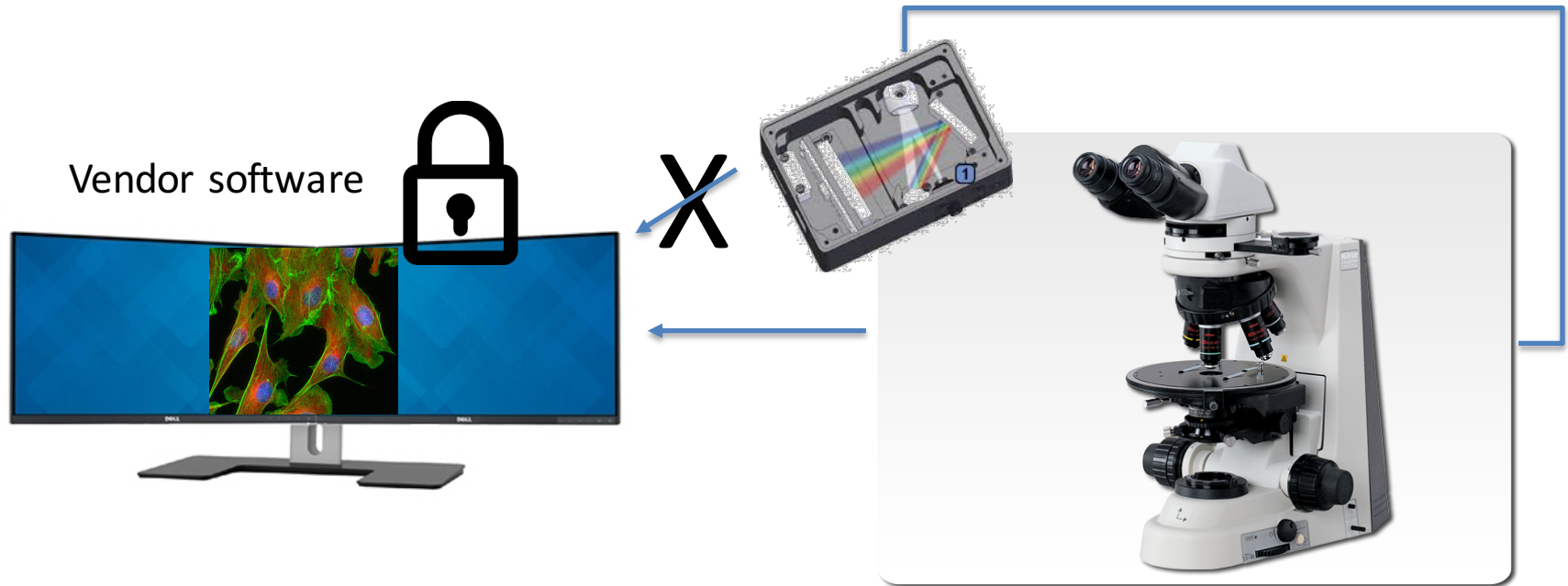
## ◆ Examples of Anaconda, WSxM

- supported by commercial entities (Continuum, Nanotec Electronica) and offered to research communities (for now)

## ◆ Role for NSRCs, National Labs, BES...?

- support projects? joint efforts?
- push vendors for low-level API's

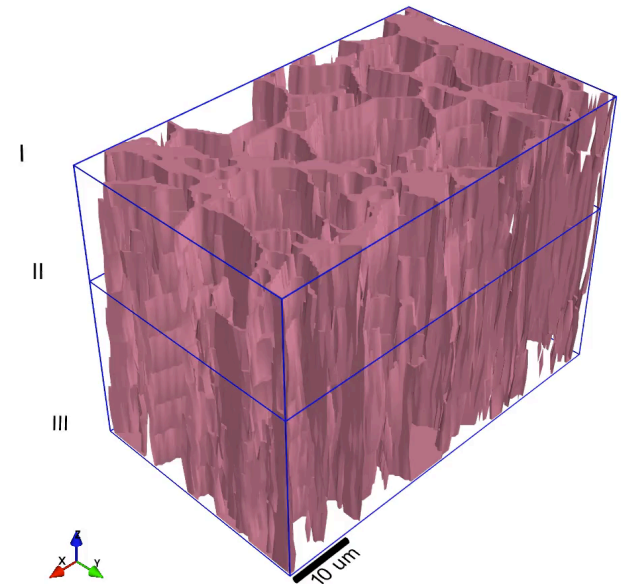
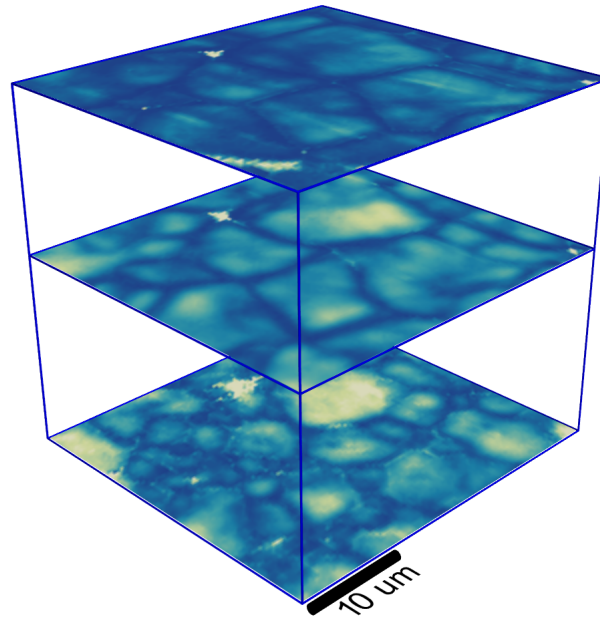
# Custom Microscope Software – Why You Need It





# Molecular Foundry Imaging Facility: 3D mapping of lifetime in solar cells

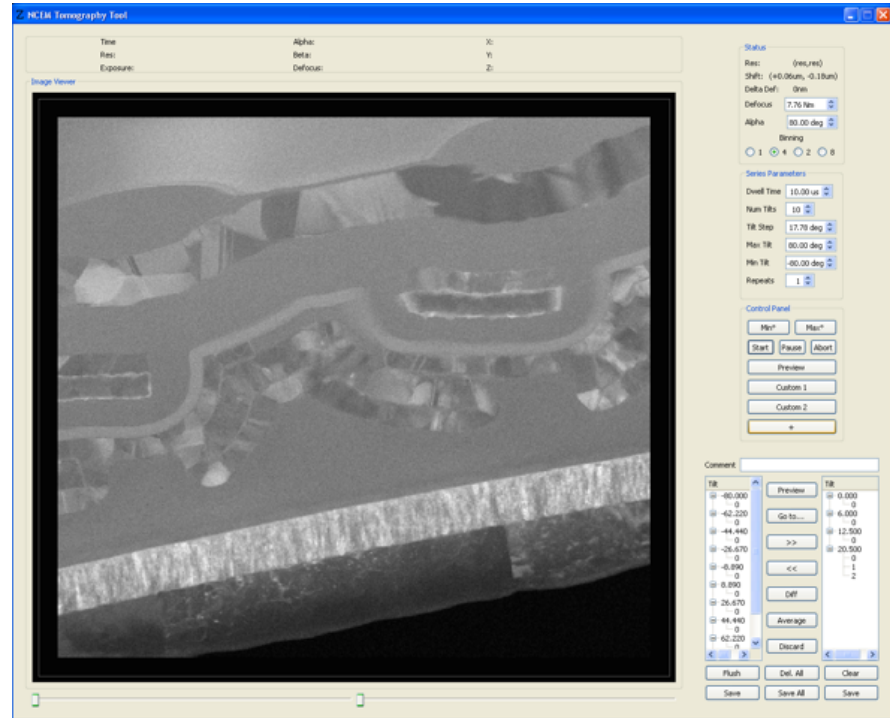
- Custom confocal Microscope
- 12 different vendor hardware pieces
- 4D (3D + t) data sets



# ScopeFoundry at NCEM



Colin Ophus  
Peter Ercius



## TEAM Microscopes

- Smart drift correction during tomography
- New imaging modalities

# Inside the Black Box: What does Microscope Software Do?



Microscope  
Software

## Takes user input

- Integration time
- For series: delta time, number of images
- For scanning: scan rate, area

## Takes measurements

- Moving stage to  $[x,y,z]$
- Measures a specified property
- Storing the value associated with  $[x,y,z]$

## \*Magic\*

- Distortion corrections
- Background subtraction

## Data visualization / Post-processing / Analysis

- Flattening
- Planarization
- Statistics

# ScopeFoundry: A custom microscopy control platform

The screenshot displays the TRPL Microscope software interface, which is divided into several functional areas:

- Left Panel (User input):** Contains various control panels for instrument settings, including:
  - 2D Scan Area:** Parameters for H, V, dh, and dv.
  - Current Position:** X, Y, Z coordinates and Move Speed.
  - AOTF Wavelength:** Freq, Power, and Freq Sweep settings.
  - Ocean Optics Spectrometer:** Acquisition and integration time controls.
  - Andor CCD:** Exposure time, EM gain, and temperature settings.
  - Power Wheel:** Encoder and Move Delta controls.
  - APD Counter:** Frequency and integration time display.
  - Power Meter:** Wavelength and power reading.
- Top Panel (Takes measurement):** Features tabs for different measurement modes (D Graph, Optimizer Graph, Andor CCD, OO Spec, Point PLE Scan, 2D PLE, PicoHarp, TRPL Map, HyperSpec, Power Scan, Photocurrent, Hardware, Measurement). It includes controls for Motorized Power Scan (Start, Interrupt) and Motorized Power Map (Start, Interrupt), along with checkboxes for APD, Spectrum, and Lifetime.
- Plot Area (Data Visualization):** Displays a large log-log plot with axes ranging from  $10^{-7}$  to  $10^{-1}$  and 0 to 100. Below the main plot are four smaller subplots:
  - Top-left: A linear plot showing a curve rising from 0 to approximately 4000.
  - Top-right: A plot showing a sharp peak at approximately 10 units on the x-axis.
  - Bottom-left: A plot showing a curve oscillating between -0.6 and 0.6.
  - Bottom-right: A plot showing a square wave or step function between 0 and 1.

# ScopeFoundry: A custom microscopy control platform

The screenshot shows the TRPL Microscope software interface. The window title is "TRPL Microscope". The interface is divided into several sections:

- 2D Scan Area:** H: 25.000 um to 26.000 um, V: 25.000 um to 27.000 um, dh: 1.0000 um, dv: 0.5000 um.
- Current Position:** X: 25.0465 um, Y: 32.3524 um, Z: 25.0267 um.
- AOTF Wavelength:** Freq: 0.00 Hz, Power: 0.
- Ocean Optics Spectrometer:** Integration Time: 0.1000 sec.
- Andor CCD:** Exposure Time (s): 1.000 sec, EM Gain: 10, Temperature: -79 C.
- Power Wheel:** Encoder: 10 steps, Move Delta: 10 steps.
- APD Counter:** 4.255190e+ Hz.
- Power Meter:** 0.00017071 W.
- Wavelength:** 1100 nm.

The main control area includes "Motorized Power Scan Controls" with buttons for "Start Motorized Power Scan" and "Interrupt", and checkboxes for "APD", "Spectrum", "Lock-in Photocurrent", "Use Shutter", and "Up and Down Sweep". Below this is "Motorized Power Scan Map Controls" with "Start Motorized Power Map" and "Interrupt Motorized Power Map" buttons.

A central plot area is highlighted with a red border. It contains the text:

Where's the magic?  
There is no magic,  
you can read, modify and understand the code

The plot area also shows several graphs: a log-log plot of intensity vs. wavelength, a linear plot of intensity vs. time, and a plot of intensity vs. position.

# ScopeFoundry: A custom microscopy control platform

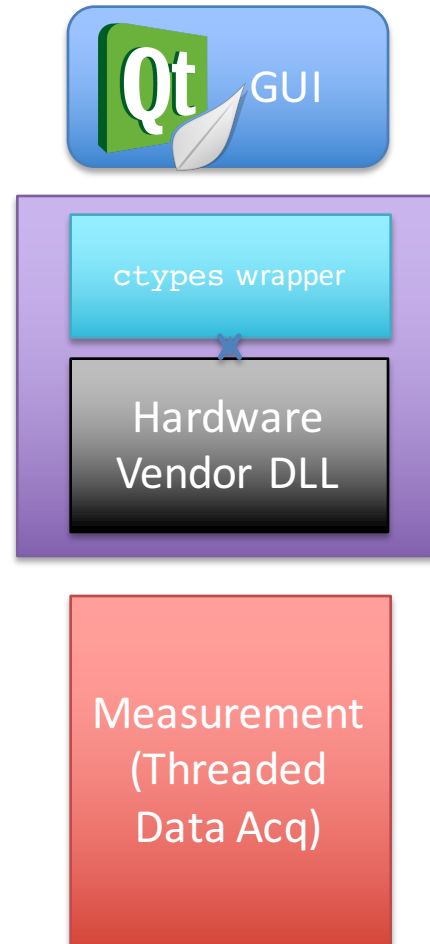
*Flexible open-source tools for microscopy and lab equipment control and data acquisition*

- Modular, multi-threaded Python GUI allows for fast data acquisition and visualization
- Rapid GUI builder with QT Creator
- Live updates of code for fast development and debugging
- Python bindings to C hardware driver APIs

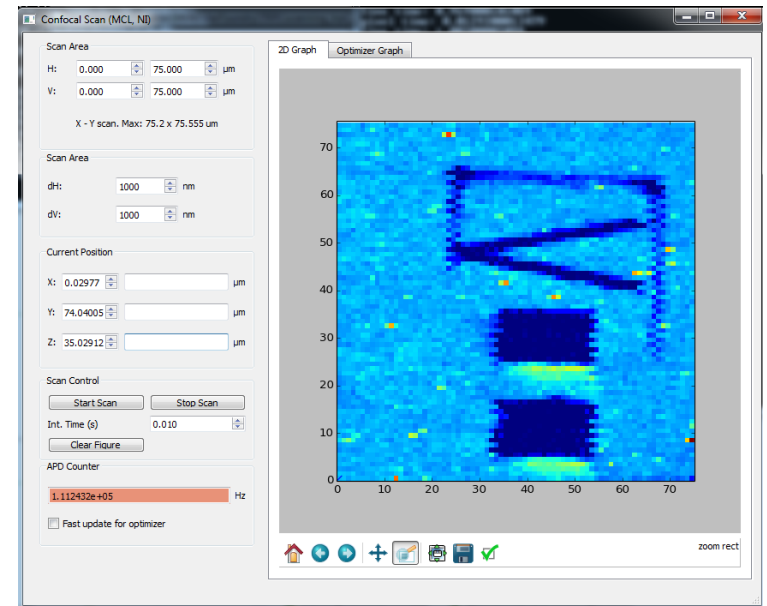
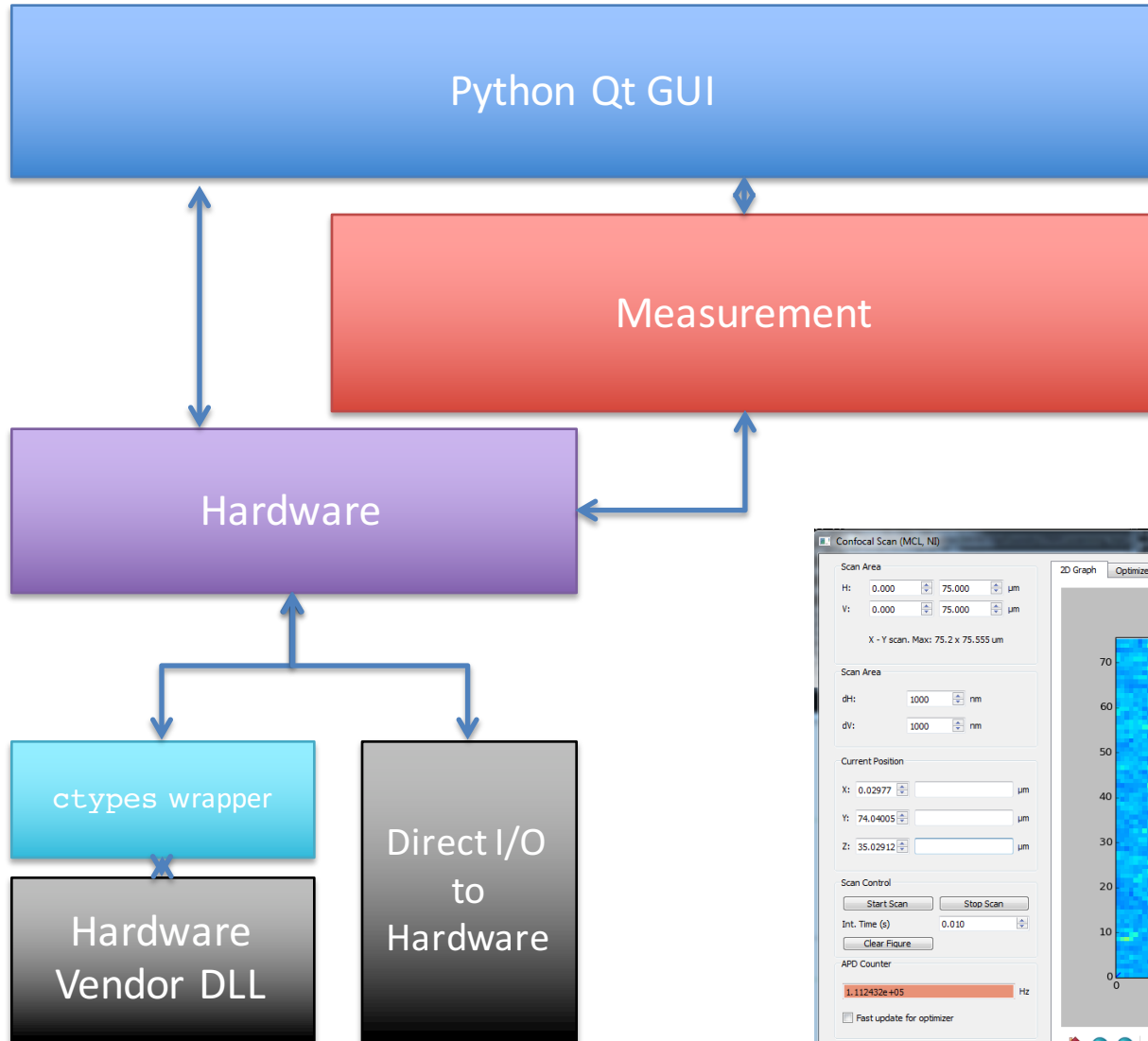


# Components Needed for Microscope Software

- Graphical User Interface
  - Plots / Visualizations
  - parameter entry: Hardware & Measurement
  - Actions – start/stop, calibrate
- Hardware
  - Wrapper for vendor supplied driver
- Measurement
  - Threaded data acquisition
    - Hardware control and coordination
    - Independent of user interface
    - Store data and write it disk
  - GUI output/visualization of data



# ScopeFoundry Modules





# Starter Interface

ScopeFoundry

Logged Quantities

Hardware

Hardware	Value
apd_counter	
connected	<input checked="" type="checkbox"/>
debug_mode	<input checked="" type="checkbox"/>
apd_count_rate	8 kHz
int_time	10 msec
dummy_mode	<input checked="" type="checkbox"/>

Logged Quantities:

dummy\_xy\_stage

connected	<input checked="" type="checkbox"/>
debug_mode	<input type="checkbox"/>
x_position	35.995 um
y_position	30.999 um

Logged Quantities:

apd\_optimizer

simple\_xy\_scan

progress	0.00 %
h0	25 um
h1	45 um
v0	25 um
v1	45 um
dh	2 um
Nh	11
Nv	20

start

interrupt

setup

setup\_figure

Settings

Hardware

Measurements

Simple XY Scan Config

APD Int Time: 0.100 sec

current X: 45.000 um

Progress: 0.00 %

H: 25.00 um  45.00 um  dh: 2.000 um  Nh: 11

V: 25.00 um  45.00 um  dv: 2.000 um  Nv: 11

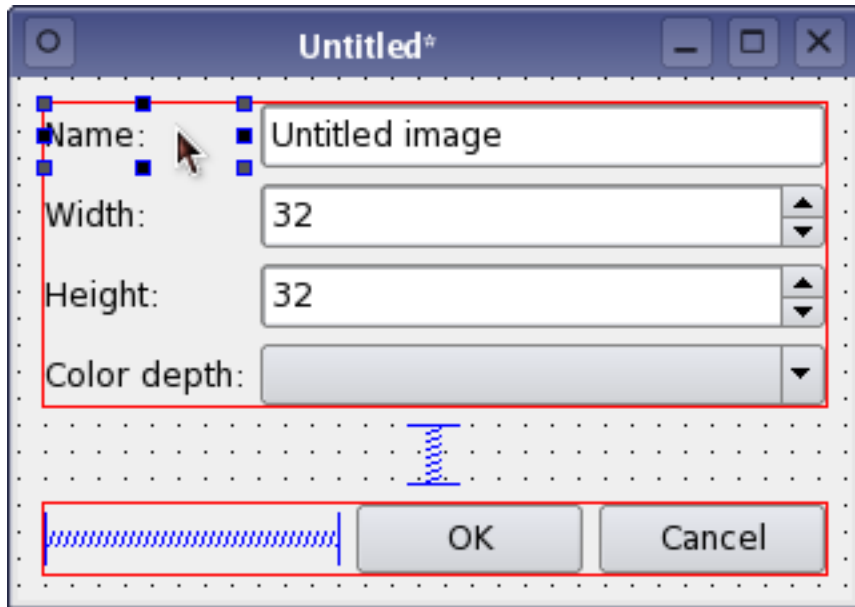
Plot

H +61.80 um [0], V +42.94 um [0]: 0.00e+00 Hz

User Designed

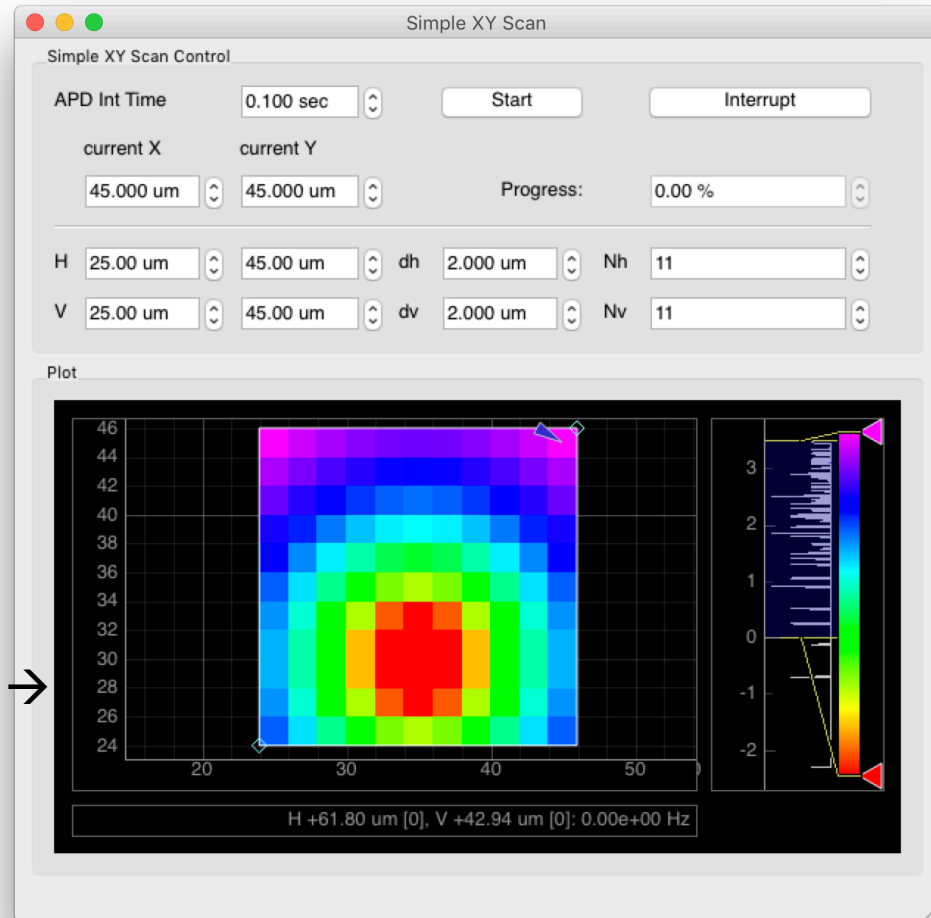
User Interface

# Custom Graphical Interface

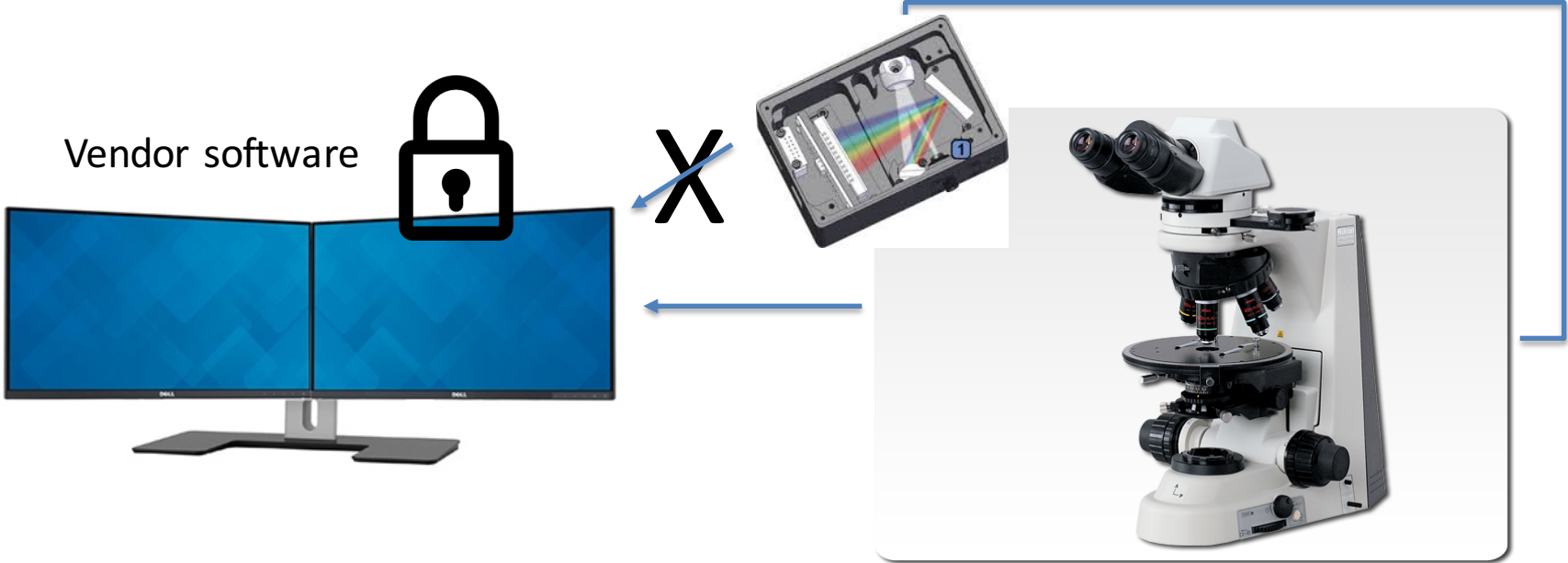


Qt Designer RAD

PyQtGraph plots →



# Power of control over your microscope

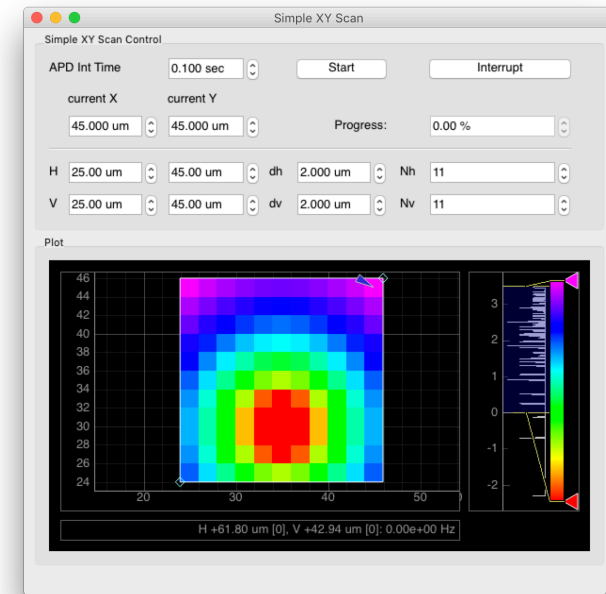


# Measurement: Simple scanning example

Threaded Run Loop:

```
for jj in range(self.Ny):  
    y = self.y_array[jj]  
    self.stage.y_position.update_value(y)  
    self.h5_file.flush() # flush data to file every line  
  
    for ii range(self.Nx):  
        self.stage.x_position.update_value(self.x_array[ii])  
        # each pixel:  
        # acquire signal and save to data array  
        self.pixel_i += 1  
        self.apd_count_rate.read_from_hardware()  
        self.apd_map_h5['data'][jj,ii] = self.apd_count_rate.val  
        spectrum = self.andor_ccd.read_spectrum()  
        self.spec_h5['data'][jj,ii,:] = spectrum[:]
```

Instant  
Hyper-spectral  
Imaging



# Demos

1. Interactive User Interface
2. In-depth online data access and control
3. Live code updates – great for debugging!

# Interactive User Interface

```
ScopeFoundry IPython Console
Jupyter QtConsole 4.1.1
Python 2.7.11 |Anaconda 2.4.1 (x86_64)| (default, Dec 6 2015, 18:57:58)
Type "copyright", "credits" or "license" for more information.

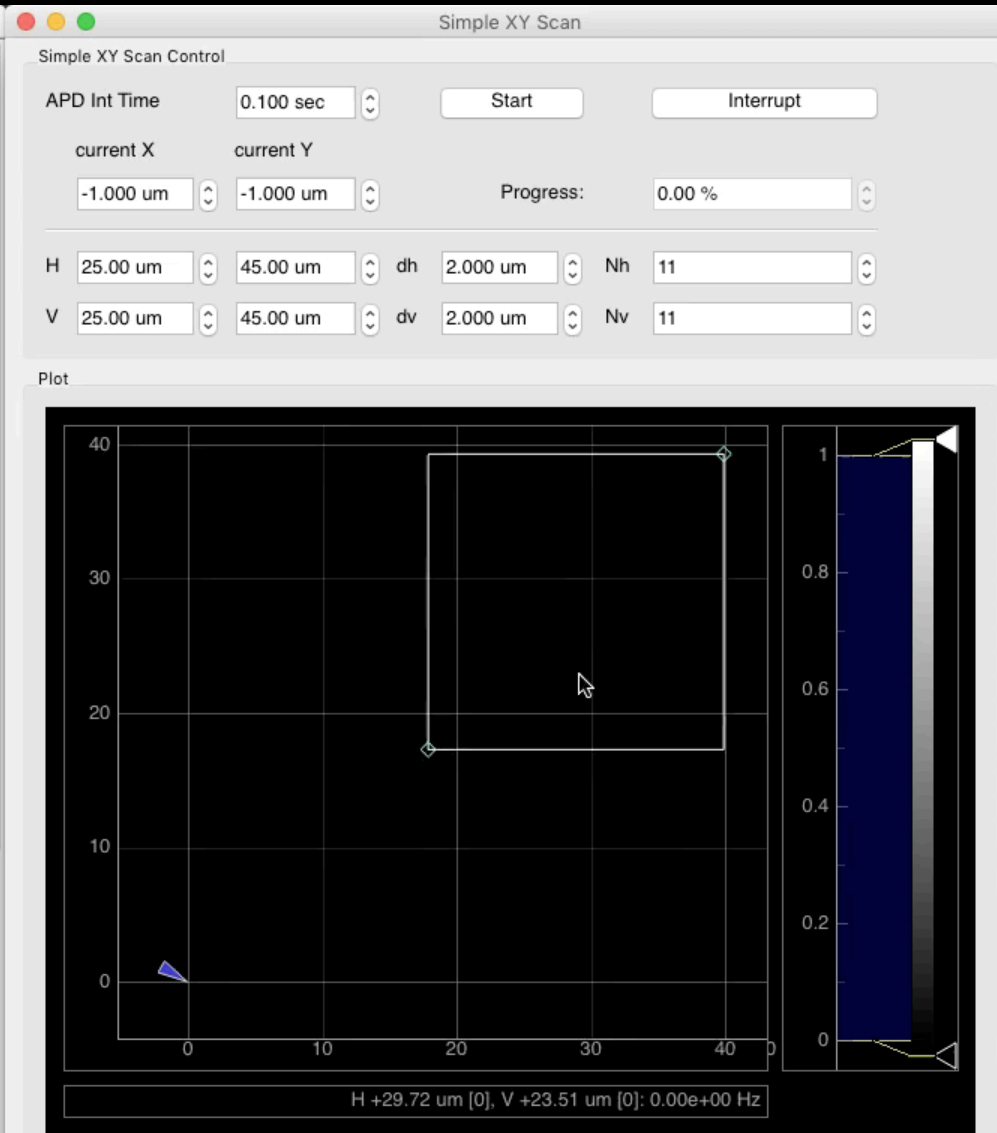
IPython 4.0.1 -- An enhanced Interactive Python.
?      -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help    -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
%gui?   -> A brief reference about the graphical user interface.

In [1]: import matplotlib.pyplot as plt

In [2]: %matplotlib inline

In [3]: xy = gui.measurement_components['simple_xy_scan']

In [4]:
```



# IPython interactive data access

The image shows a macOS desktop environment with two windows. The left window is titled "ScopeFoundry IPython Console" and contains the code `In [8]: xy.h0.update_value(10)`. The right window is titled "Simple XY Scan" and contains a control interface for a scan process. The control interface includes fields for "APD Int Time" (0.010 sec), "current X" (33.498 um), "current Y" (28.944 um), "Progress" (0.00 %), "H" (15.00 um), "V" (12.22 um), "dh" (1.233 um), "dv" (1.672 um), "Nh" (16), and "Nv" (11). Below the control interface is a "Plot" section showing a 2D heatmap with a color scale on the right. The plot has axes ranging from 0 to 50. The color scale ranges from 0 to 60. The plot shows a blue region on the left and a pink region on the right. The status bar at the bottom of the plot reads "H -10.02 um [0], V +46.90 um [0]: 0.00e+00 Hz".

python

ScopeFoundry IPython Console

```
In [8]: xy.h0.update_value(10)
```

Simple XY Scan

Simple XY Scan Control

APD Int Time: 0.010 sec [Start] [Interrupt]

current X: 33.498 um current Y: 28.944 um Progress: 0.00 %

H: 15.00 um dh: 1.233 um Nh: 16

V: 12.22 um dv: 1.672 um Nv: 11

Plot

H -10.02 um [0], V +46.90 um [0]: 0.00e+00 Hz

# Live Code Update

Eclipse IDE showing the source code for `simple_xy_scan.py`. The code includes a `finally` block for progress updates and an `acquire_pixel` method that calculates the APD count based on the current X and Y positions.

```
self.progress.update_value(100.0)
finally:
    # set all logged quantities writable
    for lname in "h0 h1 v0 v1 dh dv Nh Nv".split():
        self.logged_quantities[lname].change_writable(True)
    self.h5_file.close()

def acquire_pixel(self, pixel_i, ii, jj):
    x = self.stage.x_position.val
    y = self.stage.y_position.val
    apd_count = self.apd_count_rate.read_from_ha
    apd_count = 0.2*((x-35)**2 + (y-30)**2)
    self.apd_map[jj,ii] = apd_count
    self.apd_map_h5['data'][jj,ii] = apd_count

def clear_qt_attr(self, attr_name):
    if hasattr(self, attr_name):
        attr = getattr(self, attr_name)
        attr.deleteLater()
        del attr

def setup_figure(self):
    self.compute_scan_params()
```

Simple XY Scan Control interface showing scan parameters and a plot.

Simple XY Scan Control

APD Int Time: 0.100 sec [Start] [Interrupt]

current X: -1.000 um current Y: -1.000 um Progress: 0.00 %

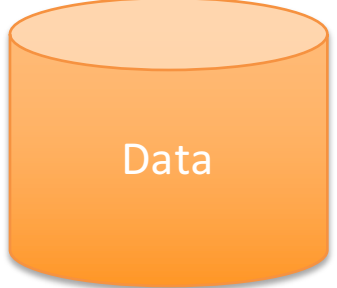
H: 7.15 um 42.22 um dh: 1.753 um Nh: 21

V: 7.57 um 42.26 um dv: 1.734 um Nv: 21

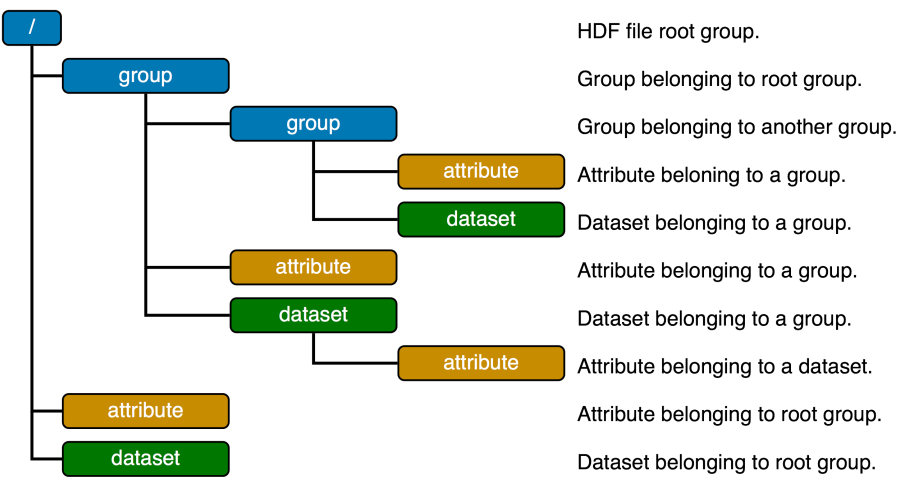
Plot

H +56.06 um [0], V +45.19 um [0]: 0.00e+00 Hz





# Standardizing data formats: HDF5



HDF file root group.  
 Group belonging to root group.  
 Group belonging to another group.  
 Attribute belonging to a group.  
 Dataset belonging to a group.  
 Attribute belonging to a group.  
 Dataset belonging to a group.  
 Attribute belonging to a dataset.  
 Attribute belonging to root group.  
 Dataset belonging to root group.

## HDF5:

Open source library for handling hierarchical data with 'attributes' (i.e. metadata)

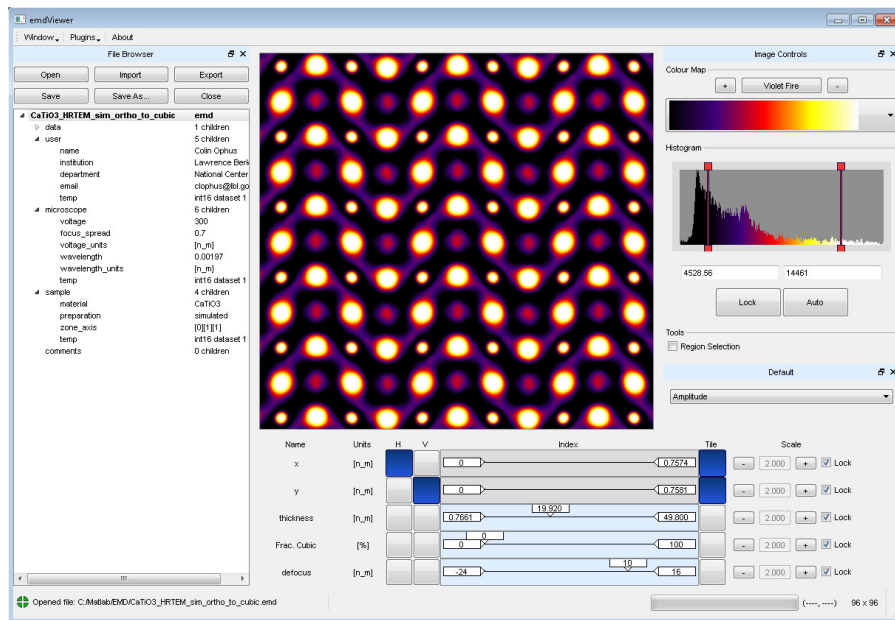
Programming language agnostic

## EMDViewer

NCEM is developing an open source viewer for N-dimensional HDF5 data

Colin Ophus

<http://emdatasets.lbl.gov/>



# Conclusions

- ScopeFoundry used in many measurement techniques at the Molecular Foundry, NCEM. Not all are scanning microscopy
- General availability soon!
- Come talk to us about using it for your experiments

Frank: [dfogletree@lbl.gov](mailto:dfogletree@lbl.gov)

Ed: [esbarnard@lbl.gov](mailto:esbarnard@lbl.gov)