

# Integrating Flexible and Efficient Image Acquisition with Open software

EuBIAS 2015

Paris

# Overview

- NIS Elements and Open Software
- NIS Elements General Extensibility
- JOBS: the experiment builder
- Illumination Sequence

# NIS Elements and Open Software

NIS Elements is strong in acquisition and general imaging:

- In addition to Nikon devices we support:
  - over 40 camera models (EMCCD, sCMOS, CCD) and
  - over 60 3<sup>rd</sup> party devices (Spinning disks, lasers, filters, stages, incubators, ...)
- General analysis and measurement (Ratio, Counting, Tracking, ...)
- 3D Segmentation and Visualization

Open Software is strong in specific imaging:

- Niche hardware support (robotic well-plate loaders)
- Implementations of novel imaging techniques (light-sheet)
- Dedicated advanced image analysis algorithms (thunderStorm)
- Statistics and Data Presentation tools (R)
- Complex data-mining pipeline (CellProfiler)

# NIS Elements General Extensibility

## Macro language

- Load functions from 3<sup>rd</sup> party DLLS (e.g. to control simple devices)
- Control RS232 devices via reading/writing to serial port
- Run other processes
  - Save Image in NIS
  - Process in ImageJ (blocking)
  - Reload in NIS

## Command line options

- NIS Elements as Slave
- Run NIS Elements from another app
  - Execute given task in NIS (blocking)
  - Continue the pipeline

```
#import ("MyModule.dll");
#import int MyFunction(LPSTR file);

char OutBuffer[1000];
char currentFilename[MAX_FILE];

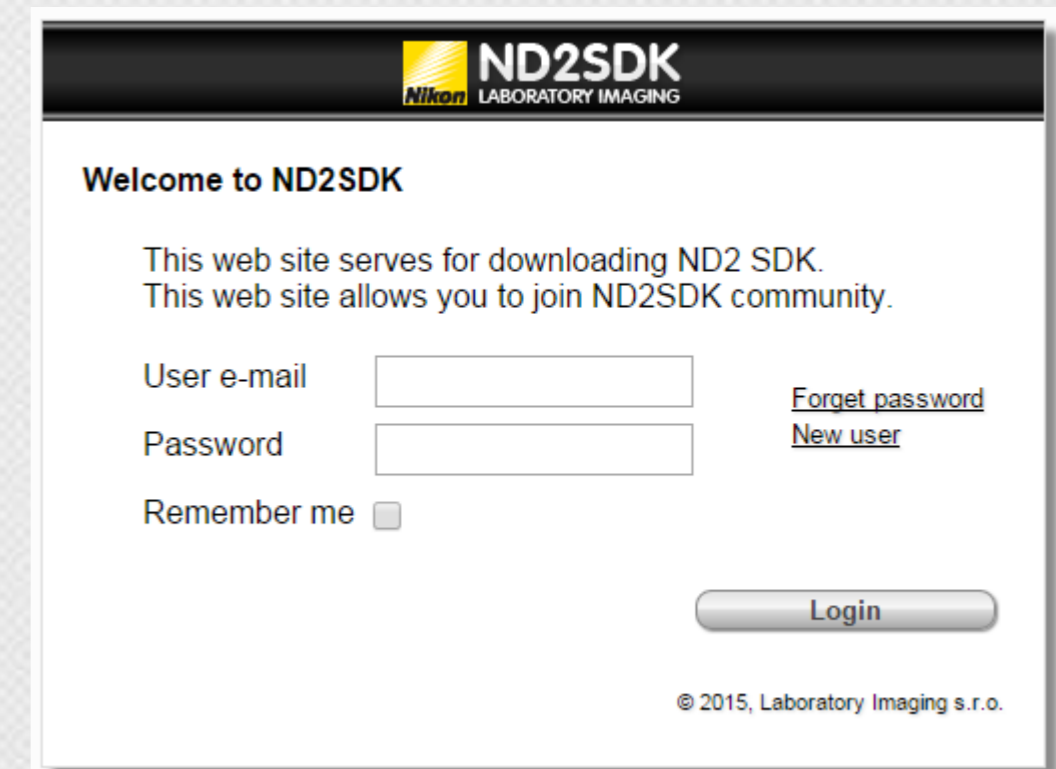
int testImportedFunc()
{
    Get_Filename(FILE_IMAGE, currentFilename);
    MyFunction(currentFilename);
    return TRUE;
}

int testRS232|()
{
    OpenPort(6, 115200, 8, "N", 1);
    WritePortEx(1, "1PA1.436", 13, 1, OutBuffer, 0);
}

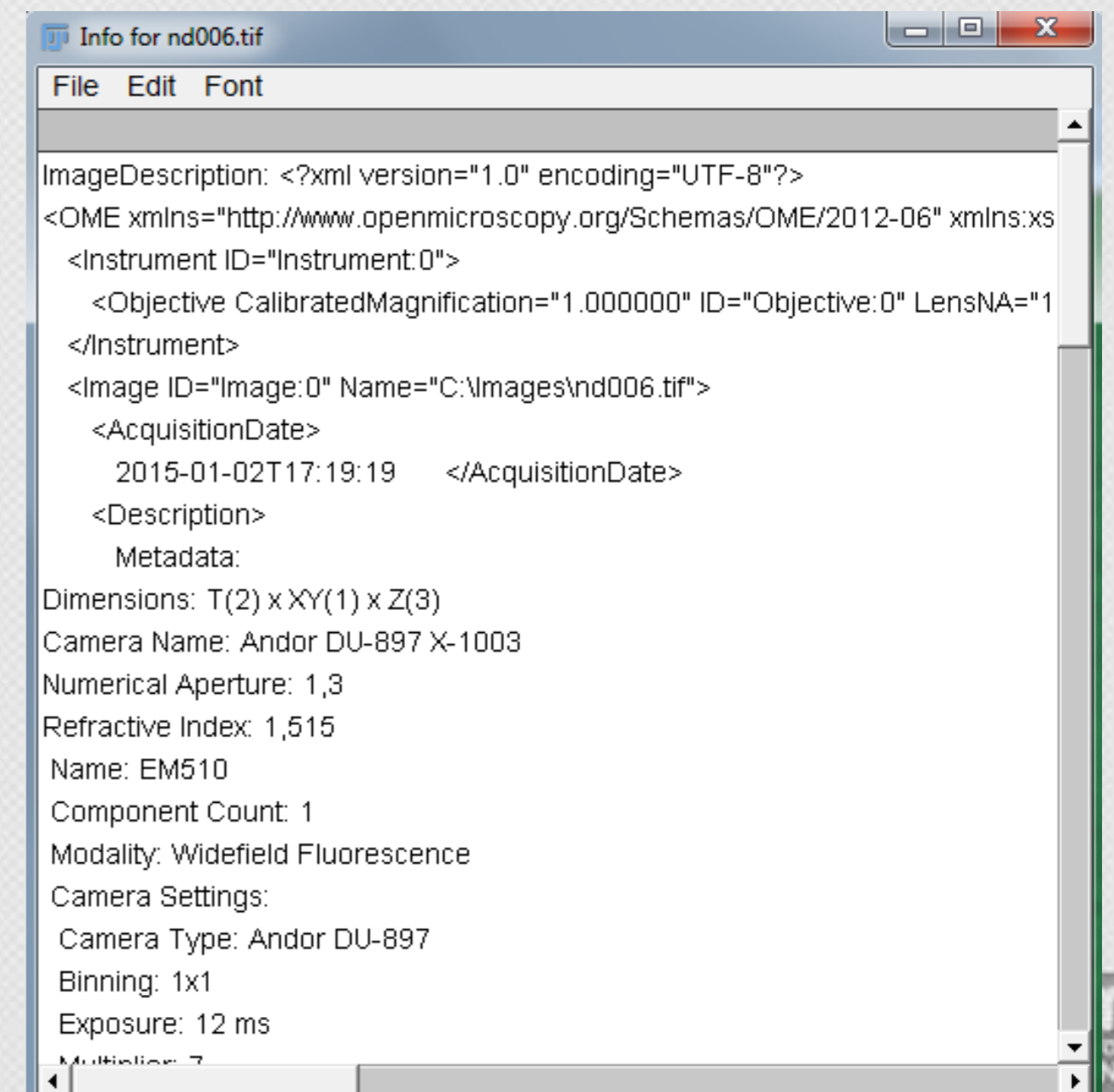
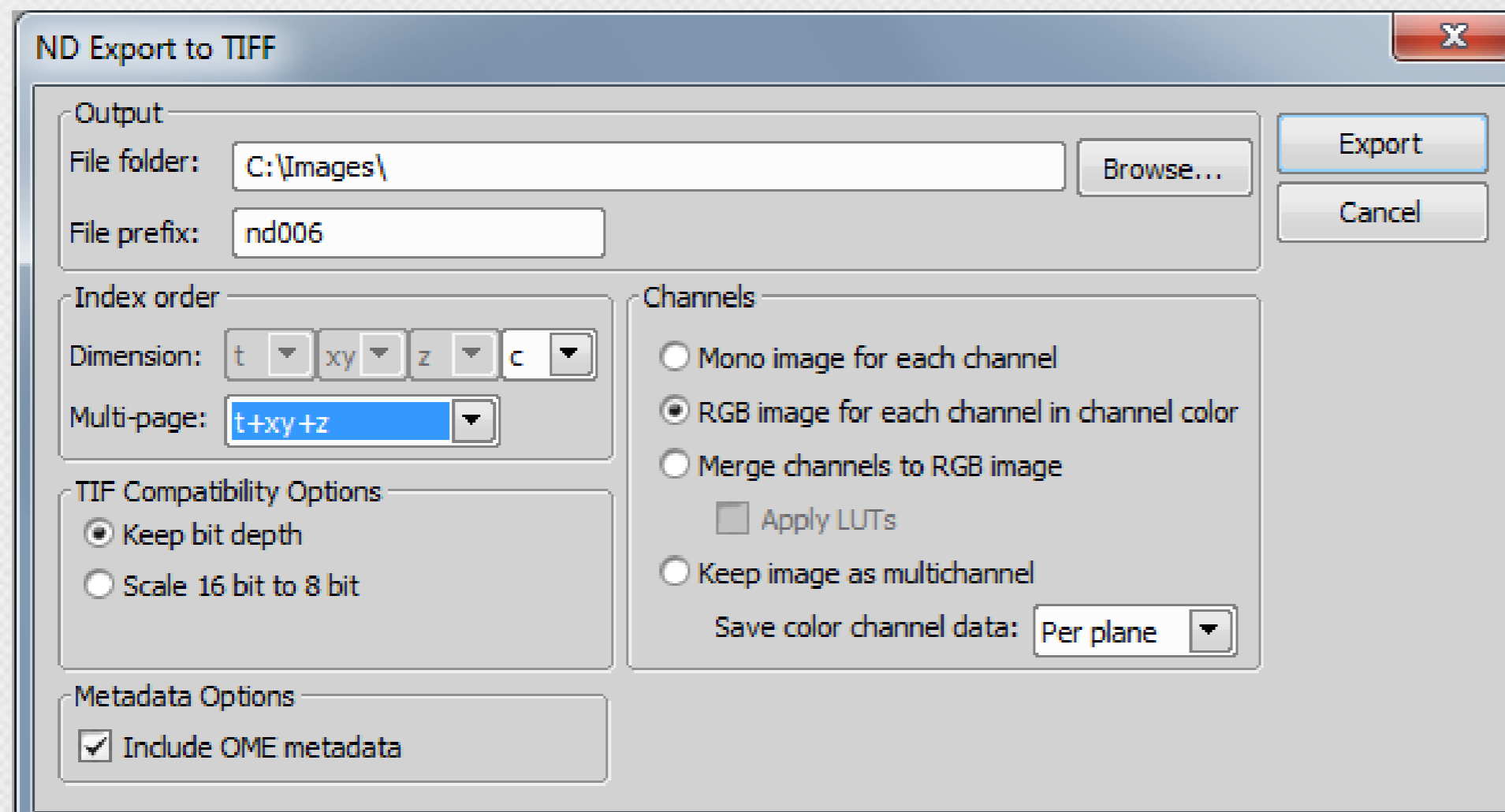
int testImageJ()
{
    ImageSaveAs("C:\images\exchange.tif", 2, 0);
    ImageClose();
    Int_ExecProgramWait("C:\Fiji.app\ImageJ-win32.exe"
        " -headless -macro C:\macros\nis.ijm");
    ImageOpen("C:\images\exchange.tif");
    return TRUE;
}
```

# NIS Elements General Extensibility

- ND2 File Format SDK  
Register and download from:  
<http://www.nd2sdk.com/>



- OME TIFF export



# JOBS

HCS and Live-cell imaging bring new challenges in terms of flexibility.

NIS Elements face them with JOBS

- Visual experiment building
- Non-orthogonal acquisition
- Conditional acquisition
- *Analysis during acquisition*
- *Macros during acquisition*
- *Storage as TIFFs (OME soon)*

The screenshot displays the 'Job Definition - Rar1' window in NIS Elements. The left sidebar contains several tool categories: Well Plates, Sample Holder, Time Series, Stage XY Points, Large Images, Z-Stack, and Autofocus + Focus Surface. The main workspace is split into two panes. The top pane, titled 'new Slide', shows configuration for 'GeneratedPoints : generate points in Slide.Slide.Slide'. It includes fields for Point Set Name (GeneratedPoints), Working Area (Whole Slide), Field Size (Actual at Run-time), Objective (2 - Plan Apo 2x), Area Restriction (Whole Area), Point Placement (Regular Pattern), Distance in X (5 mm), Distance in Y (5 mm), Scan Direction (Meander), and Frames on Border (Frame inside). A preview window shows a grid of points on a slide with a working area of 21.0 mm x 14.0 mm and 8 points. The bottom pane shows a workflow editor with steps: Objective\_4x : capture definition, Capture Frames\_in\_Slide using Objective\_4x.Definition, Find\_Rare\_Cells on Frames\_in\_Slide.Image, and a conditional block 'If ( Job.Find\_Rare\_Cells.DAPI.Count > 0 ) Then' containing Objective\_60x : capture definition, ZStackDefinition : define z-stack symmetric range: step: 0.7 μ..., and ZStack : for each Z in ZStackDefinition.ZStack, with a sub-step 'Capture ZStack\_by\_60x using Objective\_60x.Definition'.

Job Definition - NewJob4\*

Rows Icons Expert mode Import Export Properties

**Well Plates**

- Define Plate
- Select Wells
- Loop over Wells
- Label Wells
- Align Plate
- Plate Loader
- Manual Plate List
- Loop over Plates
- Move to Well
- Move to Well Center
- Move to Previous/Next Well

**Sample Holder**

- Define Slide
- Define Stage Area
- Align Slide
- Manual Slide List
- Loop over Slides

**Time Series**

- Time Lapse
- Redefine Time Lapse
- Time Sequence
- Repeat N times
- Keep Object In View

**Stage XY Points**

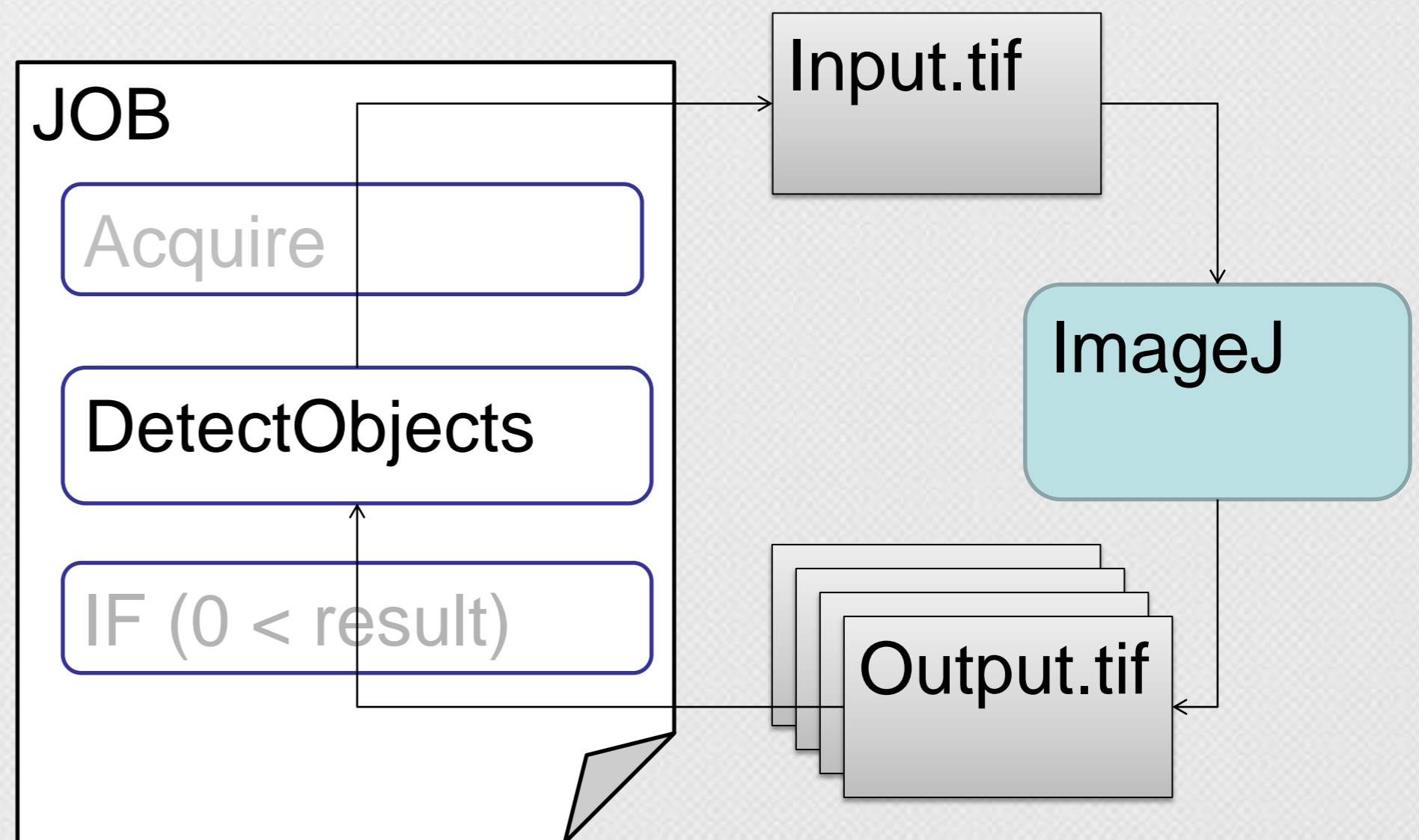
- Generate Points
- Predefined Points
- New Point Set
- Loop over Points
- Move to Previous/Next Point
- Offset Point Set
- Add Point to Point Set

Drag tasks from the Left pane and Drop them here

# The OutProc Analysis Task

A process is run on each frame.

It reads input image and writes its output mask images and text results.



**TargetPointSet**

**DetectObjects on Capture.Image**

**If** If ( 10 < Job.DetectObjects.ObjectCount ) Then

Condition: 10 < Job.DetectObjects.ObjectCount

*Objects found do something*

**Points : for each point in TargetPointSet.PointSet**

**Capture Capture1** using CaptureDefinition.Definition

Run Process For Analysis

Process: c:\Fiji.app\ImageJ-win32.exe

Process Parameters: --headless -macro "C:/outproc/nis.ijm"

Kill Process    Timeout: 600 ms



# Macros in JOBS

When Variables and Expression tasks are not enough:

- There is “Run Macro” task
- It is executed based on its position and the block it is in.
- The hardware is in defined state when it is executed.
- It can access other task parameters
- It can modify some (delays, steps, ...)

The image shows two windows from a software interface. The top window is titled 'Variables' and contains a table with the following data:

Variable Name	Type	Current Value	Always Initiate	Initial Value
Total	Integer	0	<input checked="" type="checkbox"/>	0

The bottom window is titled 'Expression' and shows the configuration for an expression task. The 'Expression Name' is 'Expression'. The 'Expression' field contains the text: `Total = Total + Job.DetectObjects.ObjectCount`. A tree view on the left shows a hierarchy of objects, with 'DetectObjects' expanded to show 'ClassName (Text)', 'TaskName (Text)', 'Binaries', 'ArraySize (Number)', 'ObjectCount (Number)', and 'Area (Number)'. An 'Operators' panel on the right contains buttons for '+', '-', '\*', '/', 'Modulo', '=', '(', and ')'. An 'Insert' button is located below the tree view.

The image shows the 'Run Macro' task configuration window. The 'Macro Name' is 'Macro'. The main area contains the following code:

```
// control your device
char OutBuffer[1000];
WritePortEx(1, "1PA1.436", 13, 1, OutBuffer, 0);

// call your dll functions
MyFunction(Job.ZStack.Current, Job.ZStack.Count,
           Job.ZStackDefinition.ZStack.Step);

// run your own program
Int_ExecProgramWait("C:\MyApp.exe");
```

The window has a toolbar with various icons for undo, redo, copy, paste, search, and other functions. The code is displayed in a monospaced font with syntax highlighting.

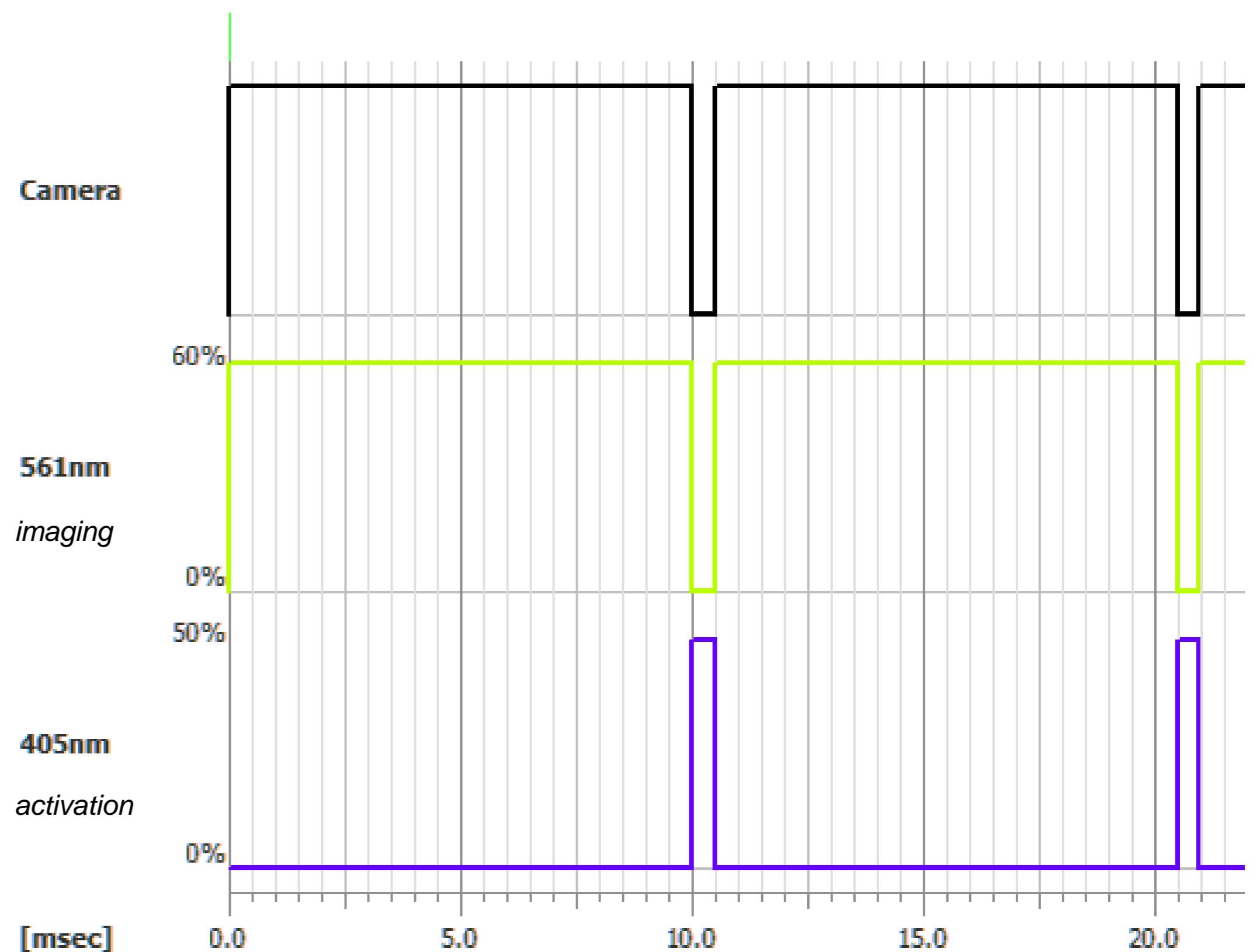
# Illumination Sequence

- Experiment driven by TTL-only signal
- Master of timing is camera FIRE signal
- Waveform generator is NIDAQ card
- Integrated devices: AOTF, LED, piezoZ, DMD

+

- ANY device with analog interface
- ANY device with TTL interface

Example: Photo-activation during transition period



# NIS Elements calculate – camera timing for

- Sensor mode:  
normal/overlapped
- Binning /ROI
- Vertical Shift speed
- Rolling/Global shutter

Figure 220.

**Flash 4.0**

sensor = overlapped, rolling shutter  
 transition = 10 msec, @ 2048 x 2048  
 exposure = 20 msec

	Cycle Definition		
Camera	Transition 10 msec	Global Exposure 10 msec	Transition 10 msec
TTL signal (Global Exposure)	Low	High	Low
Illumination	20 msec		
Cycle Time	20 msec		

Figure 221.

**iXon 897 Ultra**

sensor = normal  
 readout = 24.6 msec  
 exposure = 50 msec @ 512x512  
 Vshift = 3.3 microsec  
 Illumination mode: Follow Exposure

	Cycle Definition			
Camera	Exposure 50 msec	Transition 1.7 msec	Dead 21.2 msec	Transition 1.7 msec
TTL signal (Exposure)	High	Low	Low	Low
Illumination	74.6 msec			
Cycle Time	74.6 msec			

Figure 222.

**iXon 897 Ultra**

sensor = overlapped  
 transition = 1.7 msec  
 exposure = 1 frame, 19 msec @ 512x512  
 Vshift = 3.3 microsec  
 Illumination mode: Follow Exposure

	Cycle Definition	
Camera	Exposure 19.3 msec	Transition 1.7 msec
TTL signal (Exposure)	High	Low
Illumination	21.0 msec	
Cycle Time	21.0 msec	

# A user defines – how to

- Organize camera frames into channels
- Illuminate sample (AOTF, LED):  
follow exposure, pulsed illumination,  
during dead time, ...
- Change patterns (DMD)
- PiezoZ stacks, set PiezoZ offset,
- For a device with analog/TTL interface  
- set value at the given time-points during  
camera cycle: exposure-start, exposure-start +  
offset, exposure-end, during dead-time

Phase 1 loops 5000 Repeat

	1	2	3
Camera #1	green		
Camera #2	red		
Illumination	405, 100%		

AOTF 405nm 100 [%] Assign From Pad

Use analog power waveform. - Allow different powers for the same line during sequence.

	Cycle Definition	
	Exposure	Dead
Camera	30.00 msec	6.76 msec
TTL signal (Fire All)	High	Low
Illumination		
Cycle Time	36.76 msec	

Follow Exposure  Continuous  Custom

Repeat for: 1 Frames

Pulsed Illumination [Min Pulse Width ~ 10 µsec]

Pulse Width: 6.000 [msec]  Single

Period: 2.000 [msec]  Count: 2

Fill Exposure

Thank you...

Questions?