Micro Manager for Icy

Plan

Installation
Configuration
Acquisition

- Snap / Album
- Live
- Multi-D acquisition

Scripting / Development

- Protocols
- (Java) Script
- Plugin

Installation

By default the *µManager for Icy* plugin should be already installed. Verify than you have the last version of the plugin then launch it.

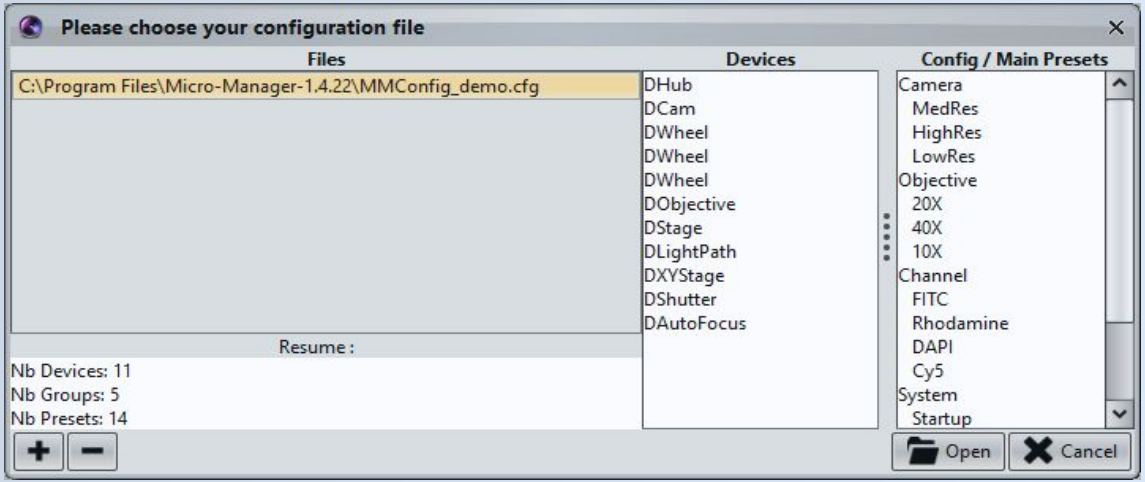
Installation

On first start you should specify the µManager's installation folder to Icy.

Be sure you installed a compatible version of µManager (currently only version from 1.4.19 to 1.4.23 are supported).

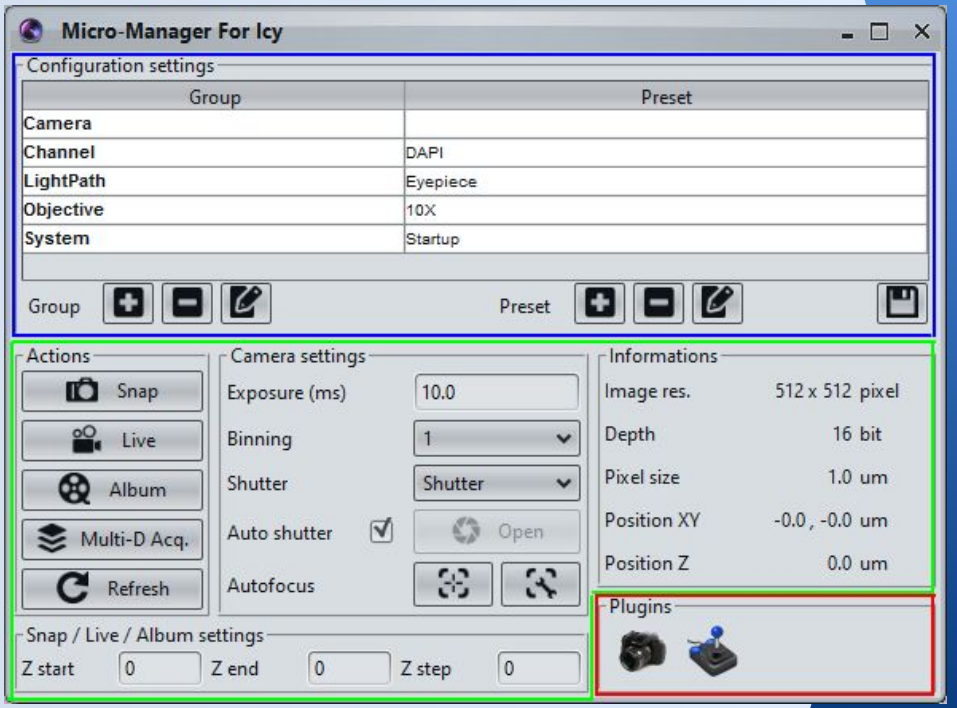
Configuration

As in µManager you should then select which configuration to load.



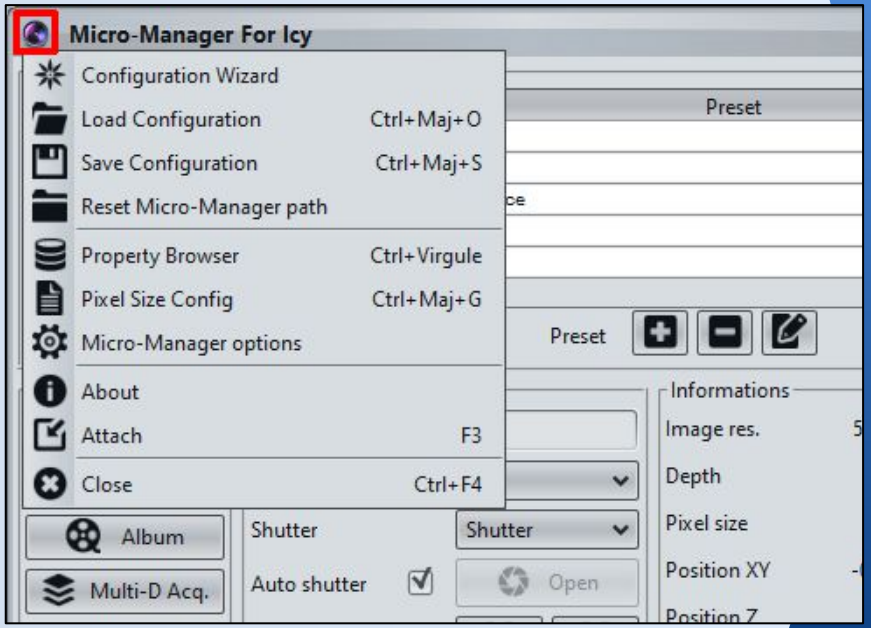
Main window

The main window is very similar to µManager with some cleanup and minors changes.
In blue we find the *Configurations Settings* part of µManager with *Groups* and *Presets*, in green we have everything about the acquisition itself (actions and settings) and finally in the red part we can find all *µManager for Icy* compatibles plugin to extend its features.



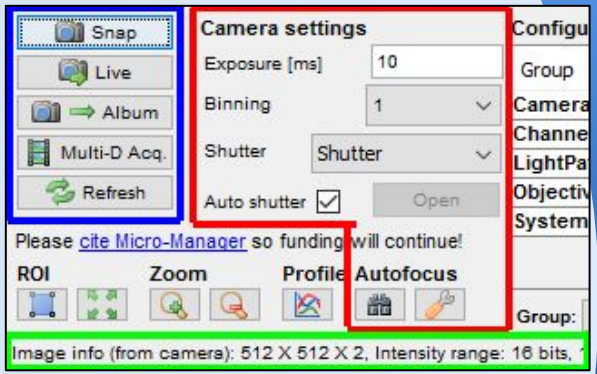
Configuration

Configuration can be done exactly as the original µManager. It's also possible to access some of the main basic features of µManager from the main window menu (by clicking on the top left icon). We can find here for instance loading / saving of settings files, the *Configuration Wizard*, the *Property Browser* and the *Pixel Size Config* of the original µManager.

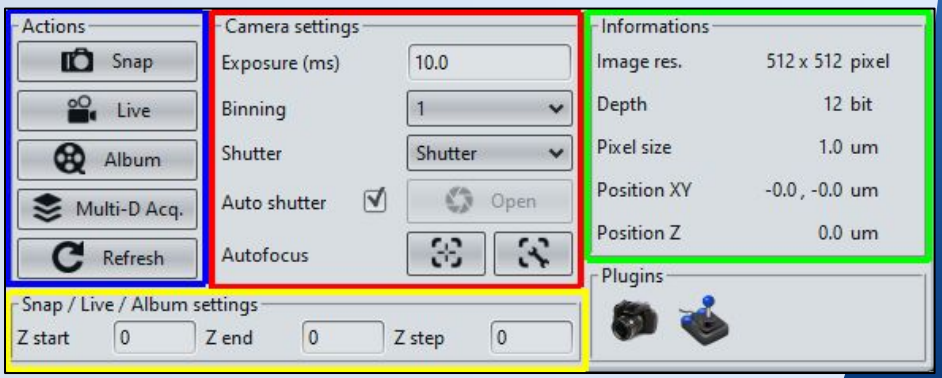


Acquisition

Acquisition part again is very close to the one from µManager. We can find here the same actions (in blue), with the same available camera settings (in red) and of course the acquisition information (in green).



The only change are these new parameters (in yellow) which allow to do live and/or basic acquisition directly in 3D.

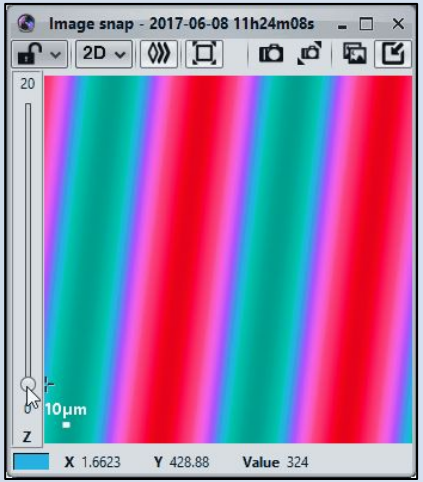


Snap / Album

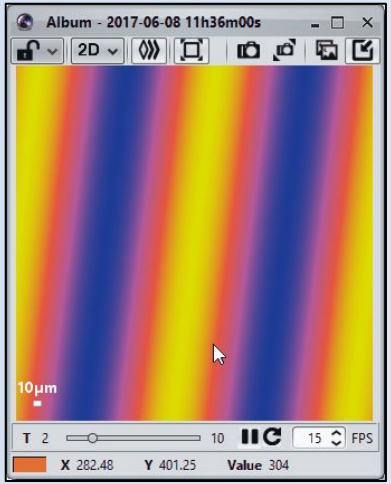
As in µManager the *Snap* operation create a new image for each acquisition where the *Album* operation will append all acquisitions in the same Sequence. As presented before and unlike the original µManager, we can now directly do 3D stack acquisition.

Snap 2D

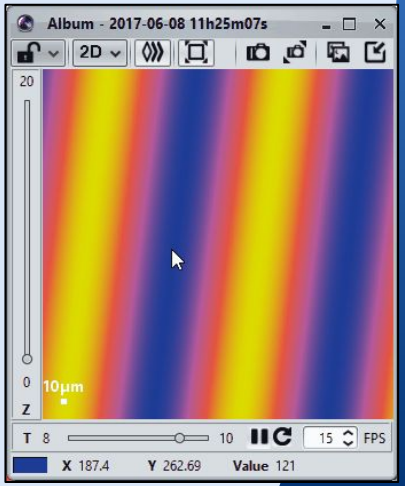
Snap 3D



Album 2D

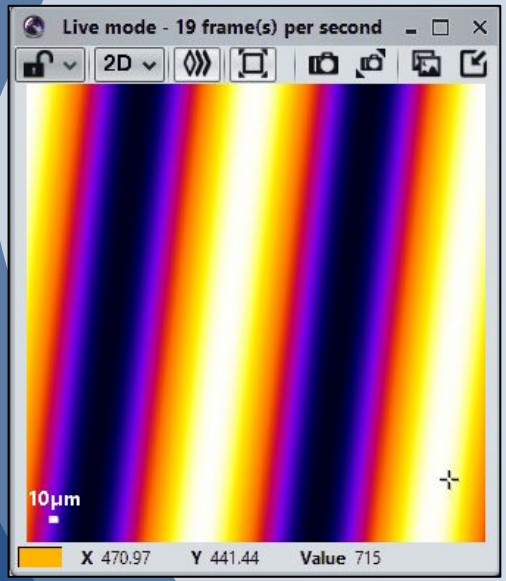


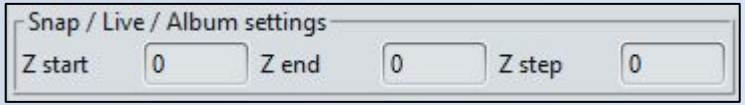
Album 3D



Live

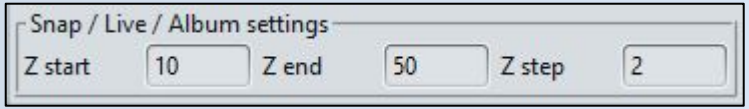
Live mode gives you a real time view from the camera as in µManager except we can now get a 3D stack view by modifying the parameters.

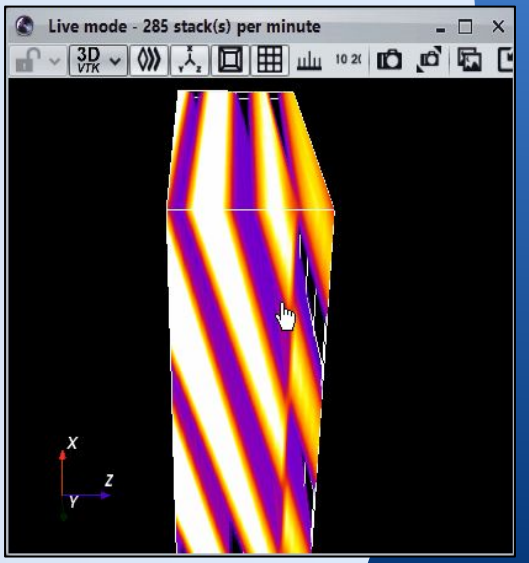




LIVE 2D

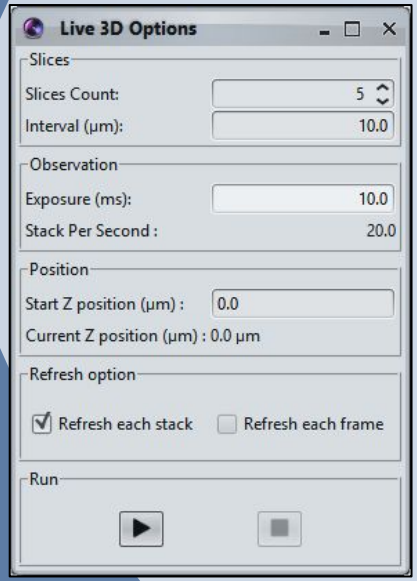
LIVE 3D

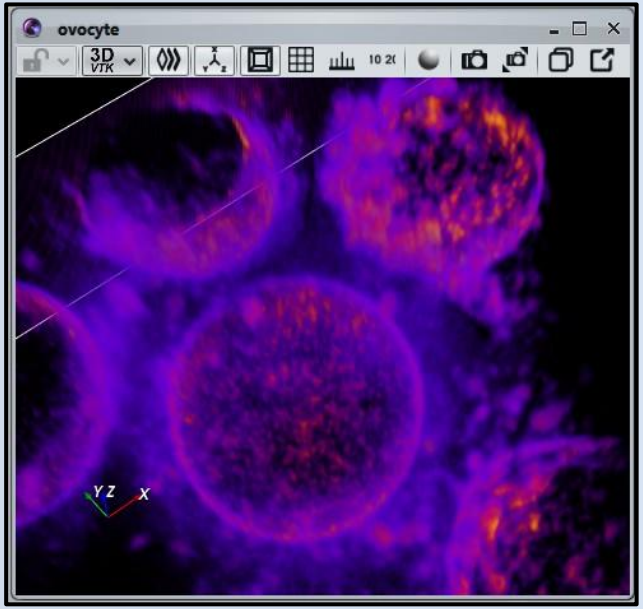


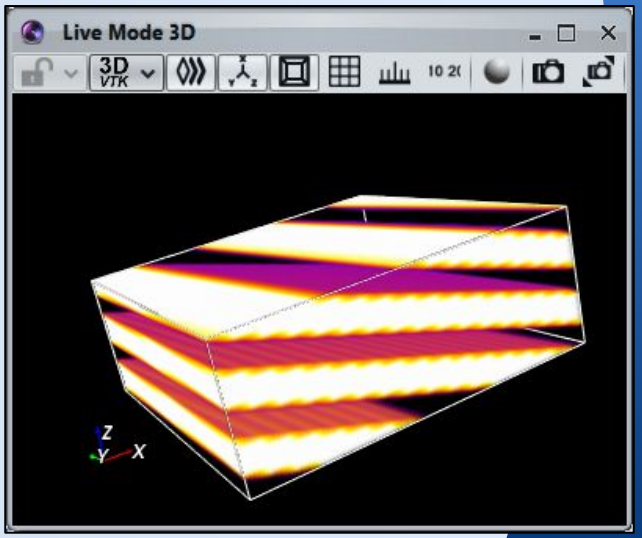


Live

The advantage of Live 3D is that it can take benefit from the 3D raycasting rendering of VTK to offer a real time 3D view.



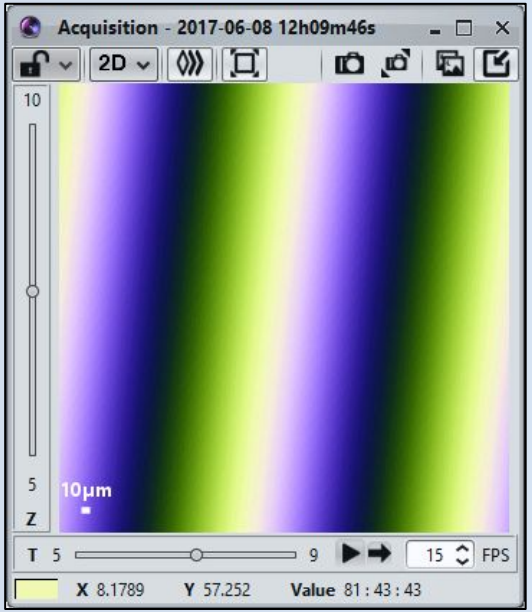




Multi-D Acquisition

This plugin correspond to the powerful *Multi-D Acquisition* tool from µManager.

The graphical interface is exactly the same as the one we can find in µManager except we can now see the acquisition progress.

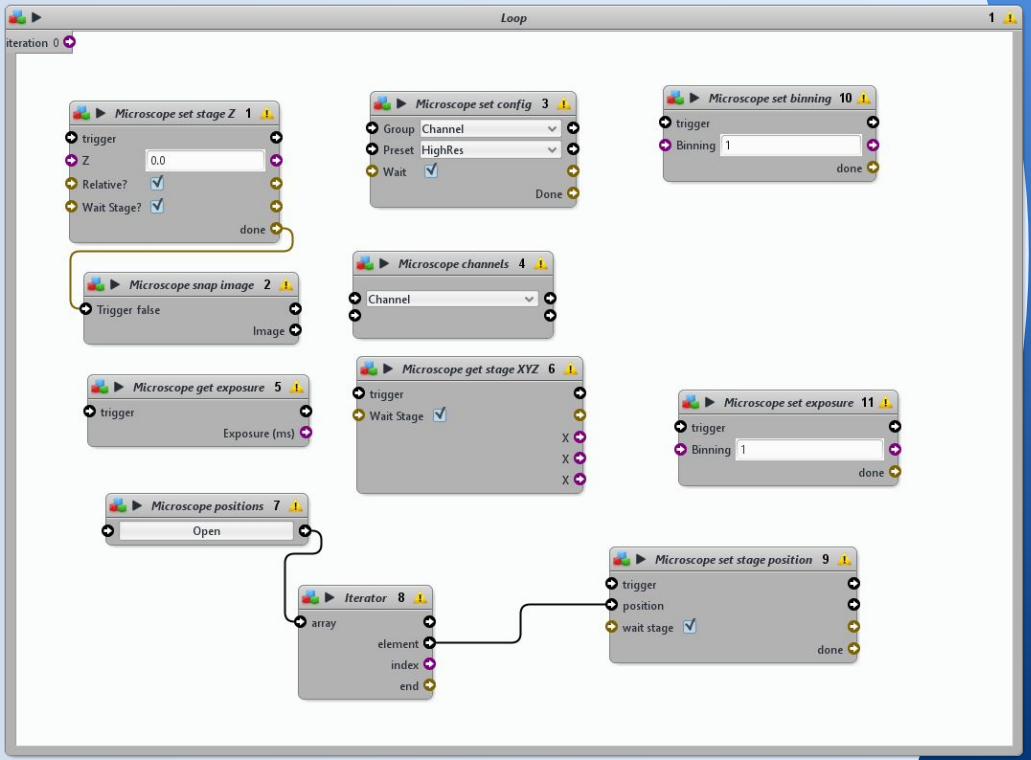


And other...

Plugin *Remote* is almost the same (except GUI) than the *Stage Position Control* from µManager (XY and Z stage position control)

Protocols

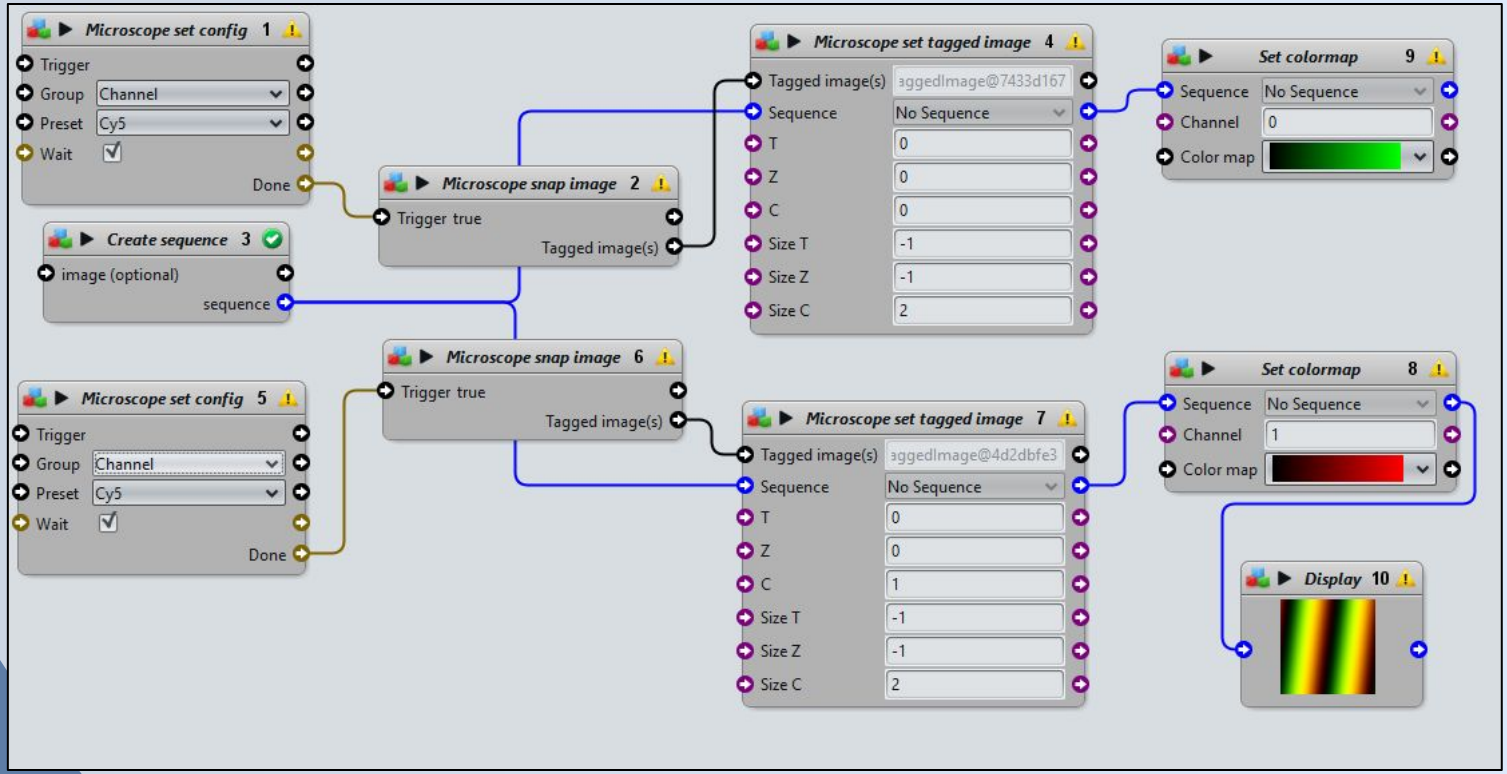
Allowing microscope control and image acquisition directly from the protocols !



Protocols - exercice 1

Goal: Design a protocol which can do an image acquisition on 2 channels.

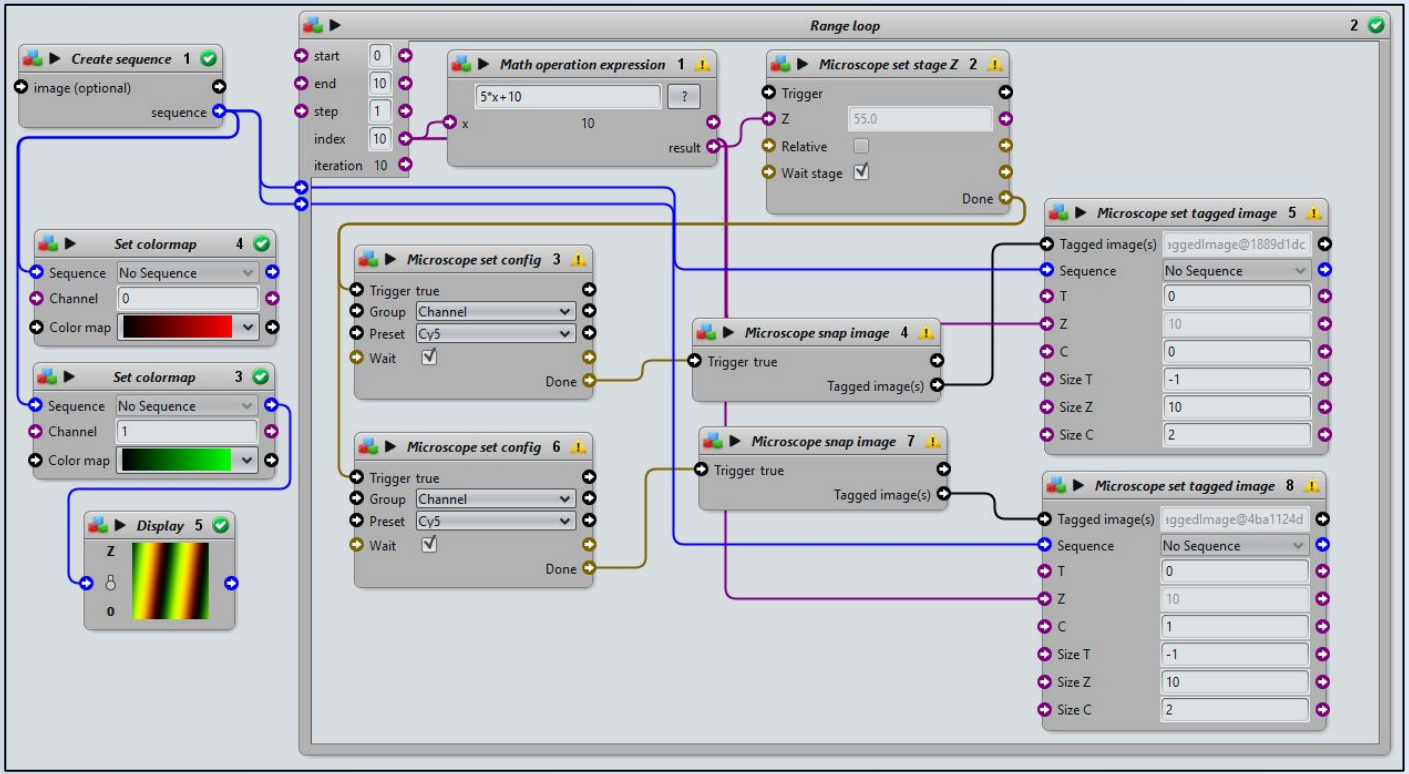
Protocols - exercice 1 - solution



Protocols - exercice 2

Goal: Design a protocol which allow to do a 3D stack acquisition. The stack should contains 10 slices where the first slice is located at Z position = 10µm then each slice are separated by 5 µm space.

Protocols - exercice 2 - solution



Script

µManager for Icy allow to control your microscope from a simple Java script.
WARNING: don't forget to launch the Micro-Manager plugin before !

This simple example will do the following operations :

- Move the XYZ stage at position [5,5,5] µm
- Snap a single image and display it

```
// move the stage to (5, 5, 5)
StageMover.moveXYAbsolute(5, 5)
StageMover.moveZAbsolute(5)

// acquire a single image
image = MicroManager.snapImage()

// create a sequence and display it
sequence = new Sequence(image)
gui.addSequence(sequence)
```

Script / Development

The main classes and methods to know to use the µManager API in Icy

MicroManager	classe principale de µManager pour Icy
StageMover	classe outil pour gérer le positionnement du microscope

MicroManager.getCore()	Get access to µManager core
MicroManager.setExposure(...)	Change camera exposure
MicroManager.snapImage()	Acquire an image and return it
MicroManager.getMetadata()	Retrieve meta data of the last acquired image
MicroManager.startLiveMode()	Start continuous acquisition mode (live)
MicroManager.stopLiveMode()	Stop continuous acquisition mode (live)
MicroManager.startAcquisition(...)	Start multiple acquisition
MicroManager.stopAcquisition(...)	Stop multiple acquisition
MicroManager.getAcquisitionResult()	Retrieve result from multiple acquisition

Script - exercice 1

Goals:

- Move the microscope stage (x,y,z) to [5,5,5]
- Acquire 3 images
- Move the stage of 10µm in Z between each acquisition
- Display the result in Icy as a 3D stack image.

Script - exercice 1 - solution

```
z = 0
sequence = new Sequence()                    // create the result sequence

StageMover.moveXYAbsolute(5, 5)              // move to position (5, 5, 5)
StageMover.moveZAbsolute(5)

image = MicroManager.snapImage()             // acquire 1 image
sequence.setImage(0, z++, image)             // set it in resulting sequence at position 0
StageMover.moveZRelative(10)                 // shift microscope Z position by 10
image = MicroManager.snapImage()             // acquire 1 image
sequence.setImage(0, z++, image)             // set it in resulting sequence at position 1
StageMover.moveZRelative(10)                 // shift microscope Z position by 10
image = MicroManager.snapImage()             // acquire 1 image
sequence.setImage(0, z, image)               // set it in resulting sequence at position 2

gui.addSequence(sequence)                    // show the sequence in Icy
```

Script - exercice 2

Goals:

- Acquire a Sequence with :
  - 20 frames with exposure time = 10 + (frame_index * 5)
  - 10 slices with Z position = -5 + slice_index
- Do an automatic threshold on obtained Sequence and display the result as a ROI (Region Of Interest)

Tips: Automatic threshold can be done using the KMeans method to retrieve the threshold intensity, then we process the threshold itself based on this intensity value (KMeans.computeKMeansThresholds(...) and Thresholder…)

Script - exercice 2 - solution

```
sequence = new Sequence()                                   // create the result sequence
gui.addSequence(sequence)                                   // display it

StageMover.moveXYAbsolute(5, 5)                             // move microscope to position XY (5, 5)

for(t = 0; t < 20; t++)
{
   MicroManager.setExposure(10 + (t * 5))                   // set exposure depending T position

   for(z = 0; z < 10; z++)
   {
         StageMover.moveZAbsolute(-5 + z, true)             // set microscope Z position by 10
         image = MicroManager.snapImage()                   // acquire 1 image
         sequence.setImage(t, z, image)                     // set it in resulting sequence at position 0
   }
}

value = KMeans.computeKMeansThresholds(sequence, 0, 2, 256)         // find threshold value
rois = Thresholder.threshold(sequence, 0, value)                    // apply threshold and get ROIs

for(i = 0; i < rois.length; i++)                            // put ROIs on the sequence
   sequence.addROI(rois[i])
```

µManager core access

You can access the internal µManager core and so get access to all the functionalities of the internal µManager API. For instance you can grab value for a specific property and more generally modify some acquisition parameters (see Programming Guide - Using device properties)

µManager core usage (from µManager):

```
core.getProperty(...)
```

µManager core usage (from Icy):

```
MicroManager.getCore().getProperty(...)
```

Script - µManager core access

```
importClass(Packages.org.micromanager.utils.MDUtils)

core = MicroManager.getCore()
image = MicroManager.snapImage()
meta = MicroManager.getMetadata()

println("Binning: " + MDUtils.getBinning(meta))
println("Pixel type: " + MDUtils.getPixelType(meta))

bd = core.getProperty("Camera", "BitDepth")
exposure = core.getProperty("Camera", "Exposure")
MicroManager.setExposure(10)
core.setProperty("Camera", "Binning", 2)
```

MicroscopePlugin class

When we develop a new Icy plugin for Micro-Manager it's important to extend the abstract class *MicroscopePlugin* instead of *Plugin* or *PluginActionable*. In this case it's important to respect the following rules:

- Overload the *start()* method (instead of the *run()* method)
- Overload the *shutdown()* method if some specific actions need to be done when plugin is terminated.

Using the *MicroscopePlugin* class assure that µManager will be loaded before the plugin starts, also it provides methods as *onSystemConfigurationLoaded()*, *onCorePropertyChanged()* and *onExposureChanged()* to detect configuration changes from µManager.

Events

Micro-Manager for Icy adds new events to make life easier for developer.

MicroManager.addAcquisitionListener(...)
Allow to listen acquisition events (start / new image / end).
MicroManager.addLiveListener(...)
Allow to listen events for Live mode (start / new image / end).

StageMover.addListener(...)
Allow to listen events from the microscope stage position (position changed)

So the developer can, for instance, easily start a specific task when receiving a new image during the acquisition.

Plugin - Tutorial project 1

```
public class MyPlugin extends MicroscopePlugin {
        @Override
        public void start()
        {
                try {
                        Sequence result = new Sequence();          // Create the resulting sequence
                        StageMover.moveXYAbsolute(5, 5);           // Set microscope X and Y positions
                        StageMover.moveZAbsolute(5);                       // Set microscope Z position
                        result.addImage(MicroManager.snapImage());         // Snap an image and add it to result
                        StageMover.moveZRelative(10);                      // Move the microscope by 10 µm in Z
                        result.addImage(MicroManager.snapImage());         // Snap again
                        StageMover.moveZRelative(10);                      // Move again
                        result.addImage(MicroManager.snapImage());         // Then Snap again
                        addSequence(result);                               // Finally, show the resulting sequence into Icy
                } catch (Exception e) {
                        // Eclipse will ask you to catch the exception, this is caused when we are unable to move the stage
                }
        }
}
```

Plugin - exercice

Objectifs :

- Start the *Live* mode
- Register to receive events from *Live* mode.
- For each received image, display the XY dimension in the output console.

Plugin - exercice - solution

```
public class MyPlugin extends MicroscopePlugin implements LiveListener {
        public void start() {
                try {
                        MicroManager.addLiveListener(this); // register listener first
                        MicroManager.startLiveMode();                // then start live acquisition
                } catch (Exception e) {
                        // we need to catch possible exception here on startLiveMode()
                }
        }

        public void liveImgReceived(IcyBufferedImage image) {
                try {
                        JSONObject meta = MicroManager.getMetadata();
                        System.out.println("Image size: " + MDUtils.getHeight(meta()) + " x "  MDUtils.getWidth(meta));
                } catch (JSONException e) {
                        // Exception when asked tags doesn't exist
                }
        }

        public void liveStarted() {}
        public void liveStopped() {}

}
```