

# MIST – Microscopy Image Stitching Tool

A Exemplar of Complex GUI Interactions

Michael Majurski  
NIST | ITL | SSD

# Outline

- Case Study at NIST
  - Microscopy Image Stitching Tool (MIST)
- Conditional GUI Elements
- Inter-dependent GUI Elements
- Handling Modes of Operation
- Missing GUI Elements



## WIPP

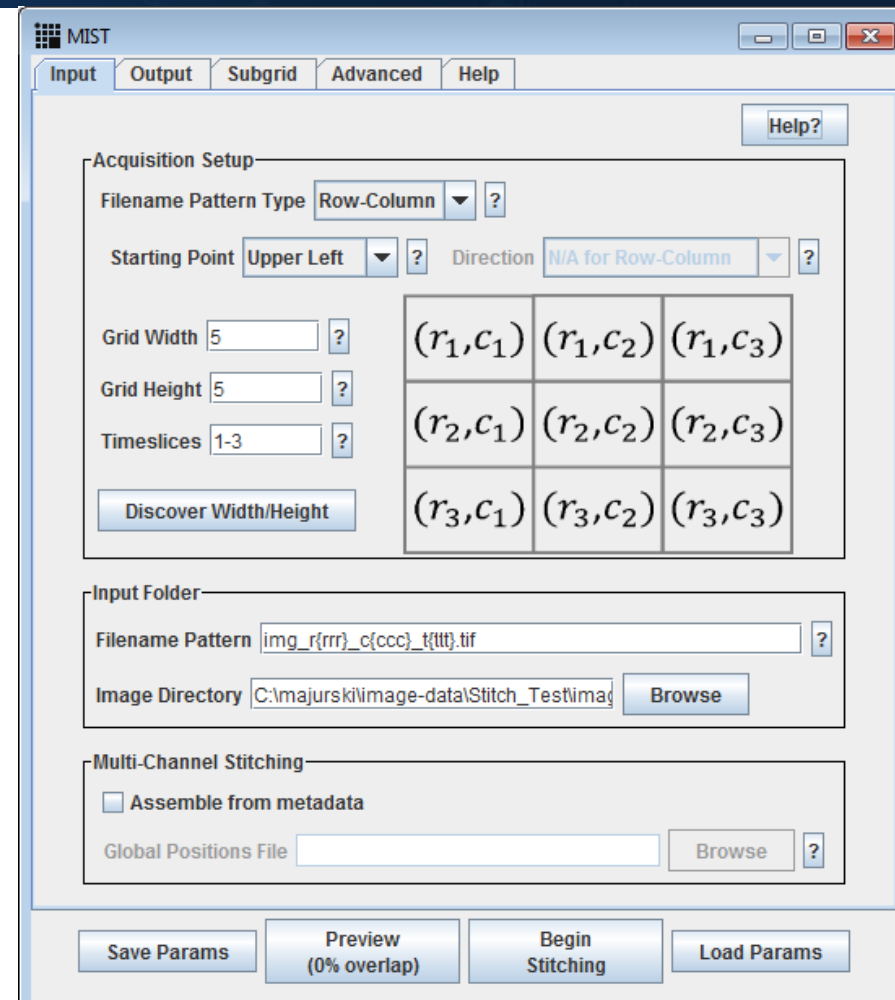
Web Image Processing Pipelines



# MIST: a GUI Case Study at NIST

- MIST stitches grids of images together
- Input needs to completely define
  - How images were acquired/organized
  - How files are named
  - Where to save results
- Input GUI controls can be:
  - Inter-dependent
  - And/or Conditional
- Multiple Modes of Operation
  - Preview
  - Stitch
  - Rebuild

2023-12-06



# Conditional GUI Elements

- Row/Column Grid Organization

- Affects Direction, Numbering
- Filename Pattern
  - `img_{rrr}_{ccc}.tif`

- Sequential Grid Organization

- Affects Direction (disabled)
- Filename Pattern
  - `img_{ppp}.tif`

Acquisition Setup

Filename Pattern Type **Row-Column** ?

Starting Point **Upper Left** ? Direction **N/A for Row-Column** ?

Grid Width  ?

Grid Height  ?

Column Start Tile  ?

Row Start Tile  ?

Timeslices  ?

$(r_1, c_1)$	$(r_1, c_2)$	$(r_1, c_3)$
$(r_2, c_1)$	$(r_2, c_2)$	$(r_2, c_3)$
$(r_3, c_1)$	$(r_3, c_2)$	$(r_3, c_3)$

Input Folder

Filename Pattern  ?

Image Directory

Acquisition Setup

Filename Pattern Type **Sequential** ?

Starting Point **Upper Left** ? Direction **Horizontal Combing** ?

Grid Width  ?

Grid Height  ?

Grid Start Tile  ?

Timeslices  ?

Input Folder

Filename Pattern  ?

Image Directory

# Inter-Dependent GUI Elements

- “Filename Pattern Type”
  - Sequential vs Row/Column
- Affects how to validate/interpret the “Filename Pattern”
- How to best capture, document, and represent these more complicated input relationships?

Acquisition Setup

Filename Pattern Type  ?

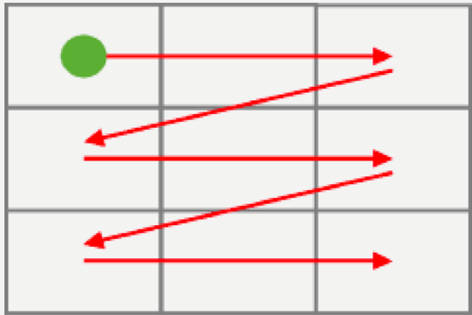
Starting Point  ? Direction  ?

Grid Width  ?

Grid Height  ?

Grid Start Tile  ?

Timeslices  ?



Input Folder

Filename Pattern  ?

Image Directory

# Tooling Modes of Operation

- Single tools can perform multiple tasks

## 1. Preview

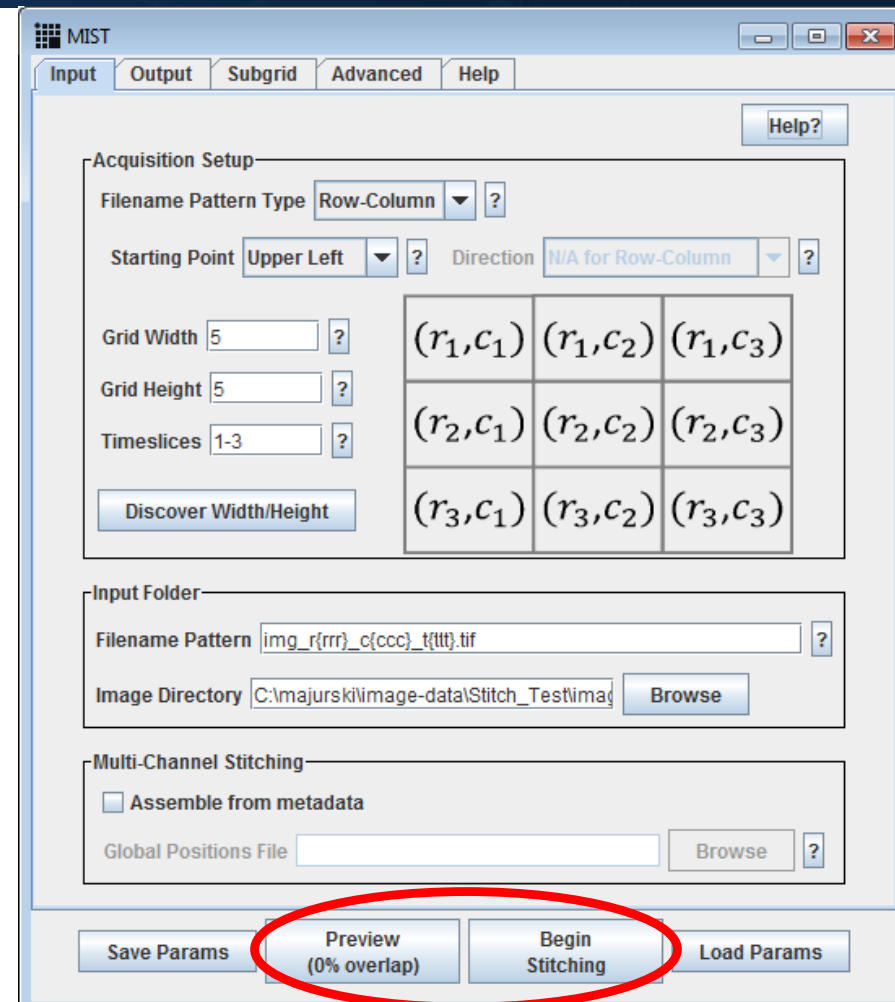
## 2. Begin Stitching

- Main “Run” operation

## 3. Assemble From Metadata

- **Hidden Mode of Operation**
- ↑ is bad practice

2023-12-06



# Current Plugin GUI Support

- Conditional Elements supported in the current Json Schema

```
{  
  "key": "inputs.startTile",  
  "title": "Start Tile: ",  
  "description": "Specify the index of the first tile (0 or 1)",  
  "condition": "model.inputs.filenamePatternType=='SEQUENTIAL'",  
},
```

- Conditional elements enable Inter-Dependent Elements
  - But for more complicated interactions, might need to allow multiple UI elements to write to the same UI "key" depending on how the UI is configured.
- Multiple Entry-Points
  - Conditional GUI elements enable basic support of tooling modes of operation, as the author can use existing parameters to modify behavior.

# Missing GUI Support

- **Explicit Execution Modes**
  - Various entry points into container
  - **Question:** Is this better left to parameter control? Or is explicit support useful?
  - Answer depends on where the complexity is placed
    - In the container argument handling?
    - In the UI logic which might call the differing container entrypoints?
- **Interactive Data Annotation**
  - How can we support containerized solutions for data annotation



- Question: Does the container interoperability care about any GUIs the containerized tool itself uses?
- NIST WIPP system has limited web-based data annotation capabilities
  - GUI is web-based with full deep zoom view
  - Annotation management/selection controls
- Data annotation tooling should exist as its own tool
  - Containerized interoperable tool control GUI elements don't need to worry about how to present any GUI elements within those tools.
  - For example: Fiji GUI has no care/control over how we organize the MIST GUI

- **MIST Case Study at NIST**
- **Conditional GUI Elements**
  - Largely supported already
- **Inter-dependent GUI Elements**
  - Marginally do-able with conditional elements
  - But lacking modeling of dependencies (is that needed?)
- **Tool Modes of Operation**
  - Clunky Support if handled via parameters
  - Is support of multiple container entry-points useful?
- **Interactive Data Annotation**
  - Bootstrapping AI models via label -> train -> refine loop
  - Containerized GUI can be tailored to the required application, and not standardized.