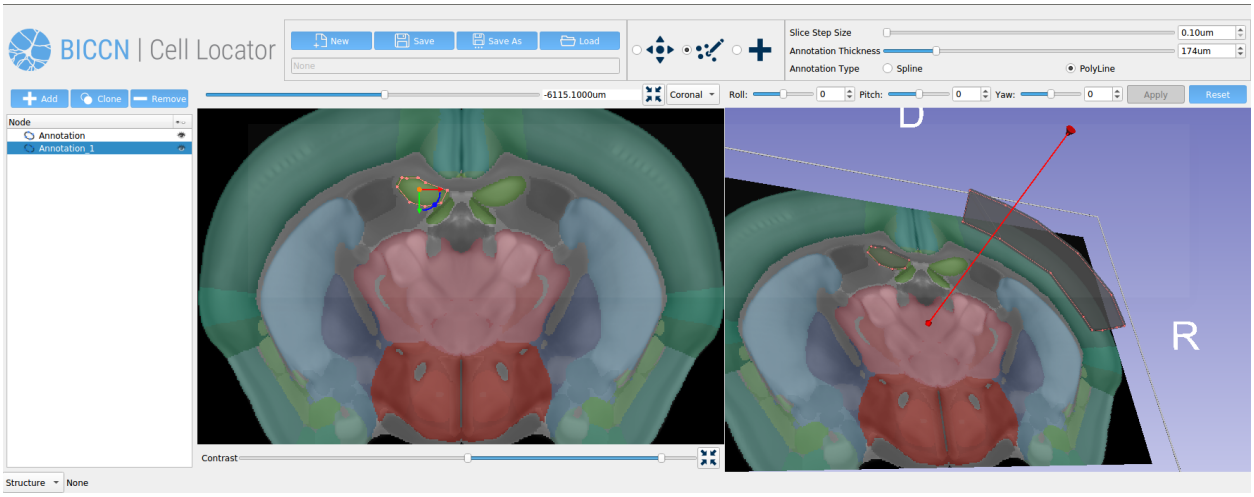
Cell Locator

Allen Institute for Brain Science, Kitware

May 10, 2023

CONTENTS

Cell Locator is a free, open source and multi-platform pinning tool to facilitate mapping samples into common 3D spaces.



A desktop application based on 3D Slicer that displays the Allen Institute for Brain Science CCF or MNI atlas, color segmented by structure of the brain, and allows a user to create a planar polyline annotation to facilitate mapping samples into the corresponding atlas.

CHAPTER

ONE

FEATURES

- Create, save, and load spline or polyline based annotation.
- Manage and save multiple annotations per file.
- Save annotations as JSON containing Spline or Polyline points as well as orientation and thickness.
- Download from / Upload to a LIMS system. See *LIMS integration*
- 3 interaction modes: Explore, Edit or Place point.
- Reslicing in arbitrary directions.
- Ontology selection: layer vs area.
- Input of Roll/Pitch/Yaw.
- Load CCF or MNI atlas. See *command-line arguments*.

1.1 Getting Started

1.1.1 Installing Cell Locator

Download the latest release on GitHub

1.1.2 System Requirements

Cell Locator is based on 3D Slicer, so refer to the 3D Slicer System Requirements.

Cross-platform Availability

Installers are only generated for 64-bit Windows, however it is possible to build Cell Locator from source on any platform which supports 3D Slicer.

Refer to the *Cell Locator build instructions*.

1.1.3 Using Cell Locator

Quick Start

After starting Cell Locator, you will choose an atlas.

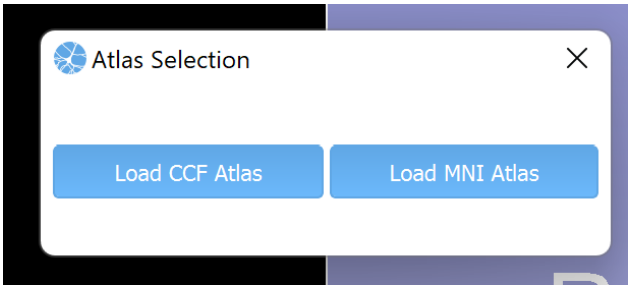
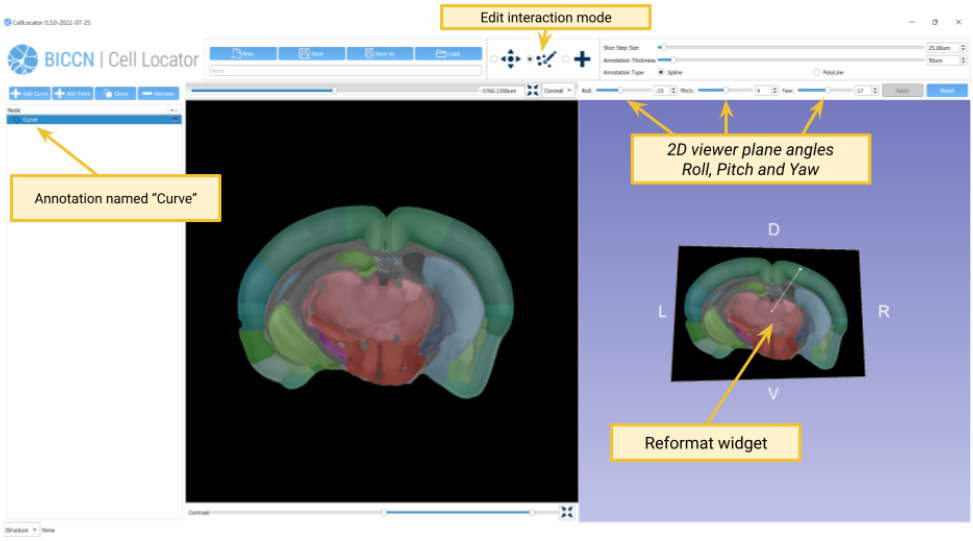


Fig. 1: Atlas Selection Startup Dialog

After selecting an atlas, the application is ready with an annotation named *Curve* already created.

Before adding points to the annotation:

1. Switch to *Edit* interaction mode.
2. Use the reformat widget available in the 3D viewer on the right to update the orientation and identify a slice plane of interest.

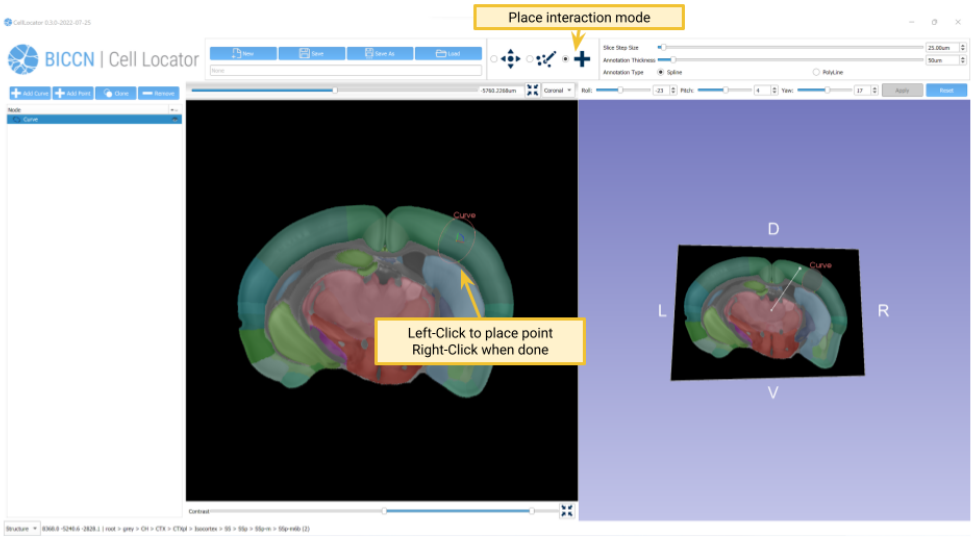


Hint

Alternatively, you may directly set the the raw, pitch and yaw angles.

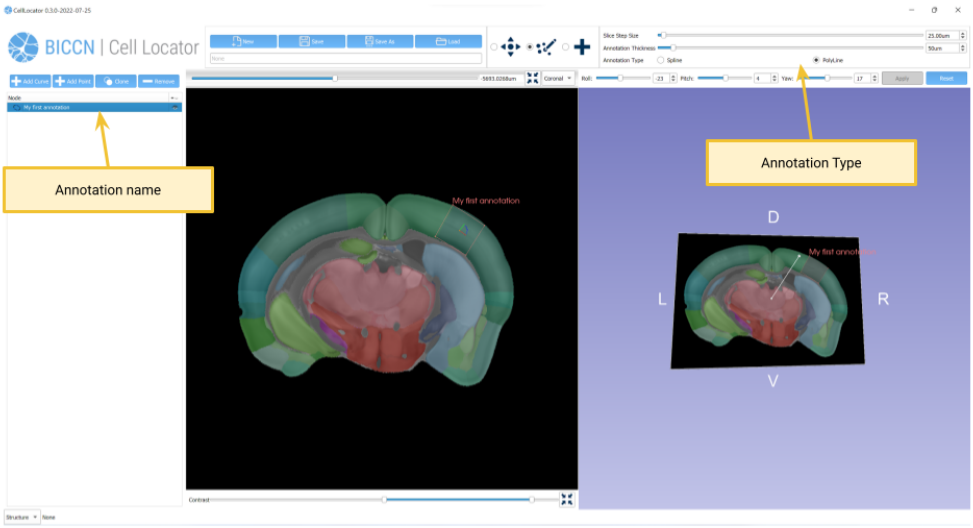
You can switch to *Place* interaction mode and start adding points to the annotation.

1. Left-Click to add (or place) points.
2. Right-Click to switch back to the *Edit* interaction mode.

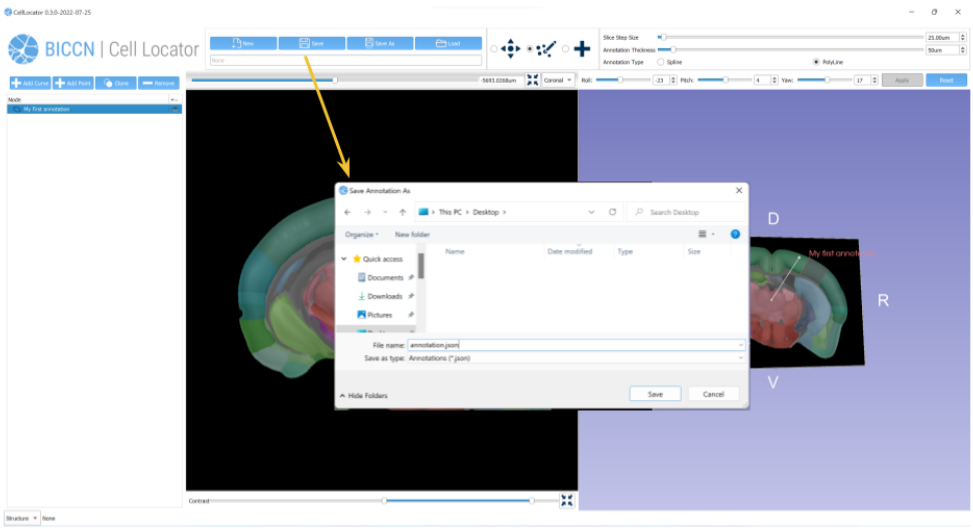


Once you are done adding points, you can update annotation properties.

1. Switch the annotation type from *Spline* to *PolyLine*.
2. Renane the extension.



Finally, you can save the annotation as a `.json` file by clicking on the *Save* button.



User manual

Browse the *Application Overview* section to learn about the application user interface.

1.2 Application Overview

1.2.1 Atlas Selection

After starting Cell Locator, the user is prompted with a dialog asking to choose which *reference atlas* to load.

Resetting Views

The `Ctrl + w` keyboard shortcut allows to reset views discarding current changes and asking to choose which reference atlas to load.

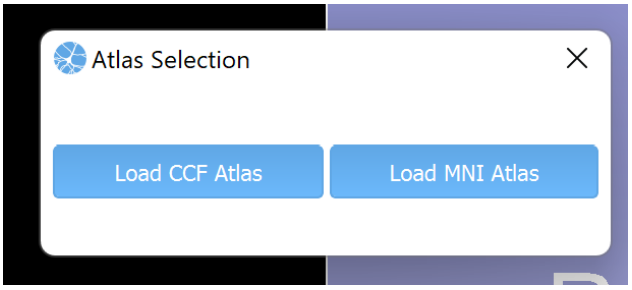
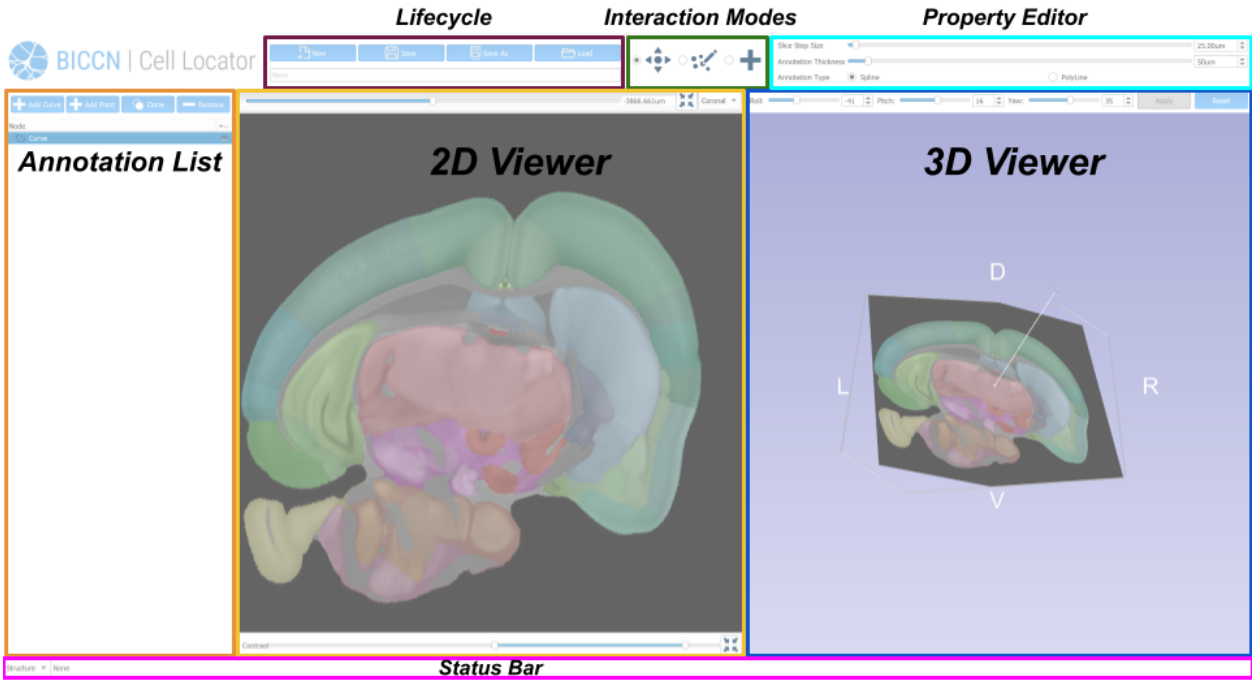


Fig. 2: Atlas Selection Startup Dialog

Automatic Atlas Selection

Starting the application specifying the `--atlas-type` *command-line argument* will skip the atlas selection startup dialog and directly load the requested altas. In this case, resetting the views also skips the atlas selection dialog.

1.2.2 User Interface



Lifecycle

- **New** allows to create a new annotation file.
- **Save** and **Save As** allow to save the current annotation(s) to file.
- **Load** allows to load annotation(s) from file.

> **Warning:** If the current annotation(s) have been modified, the user is asked if they should be saved to file before creating or loading new ones.

Annotation List

To to add, remove or duplicate an annotation, the user may click on the corresponding button:

- **Add Curve** and **Add Point** allow to add an annotation.
- **Clone** allows to duplicate the selected annotation.
- **Remove** allows to delete the selected annotation.

To rename an annotation, the user may Double-`Left-Click` on its name.

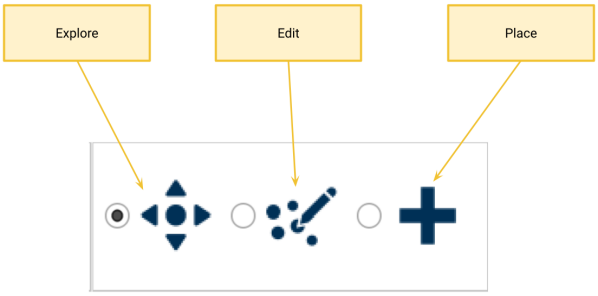
To show or hide an annotation in both viewers, the user may click on the eye icon.

Note:

- A `Curve` annotation is expected to have multiple points and the corresponding type can be set using the *Property Editor*.

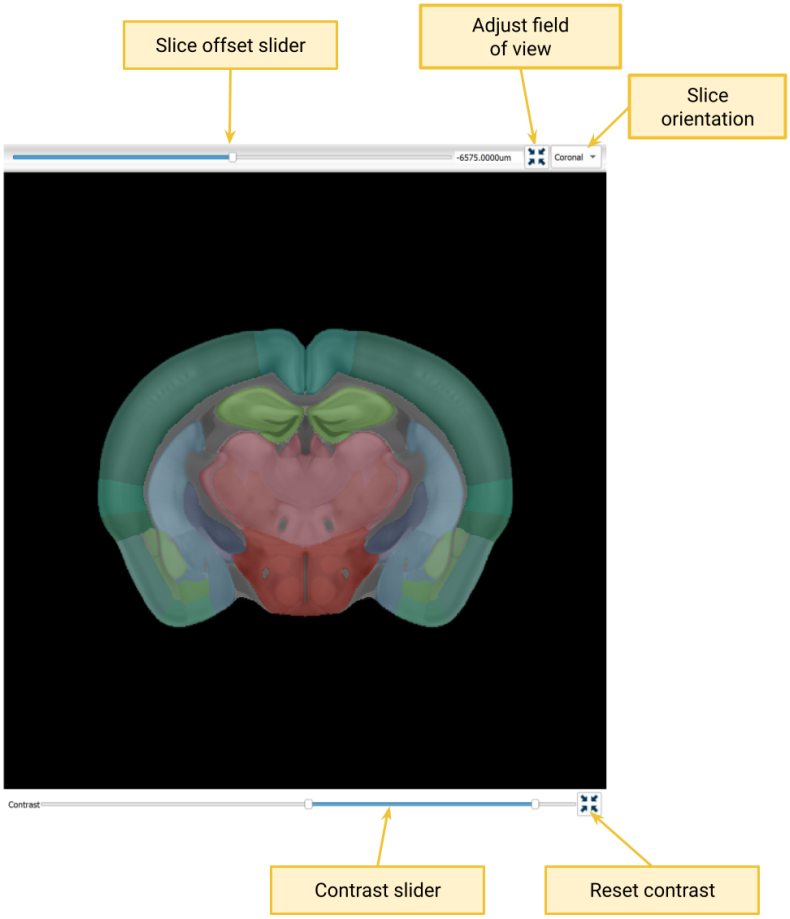
Interaction Modes

- **Explore** mode allows to interact with the viewers without updating the orientation of the currently selected annotation.
- **Edit** mode allows to update the selected annotation.
  - Position and size can be updated in the *2D Viewer*.
  - Orientation can be updated in the *3D Viewer* by using the reformat widget or by setting specific angles.
- **Place** mode allows to add point(s) to the currently selected annotation by clicking one the slice displayed in the *2D Viewer*.



2D Viewer

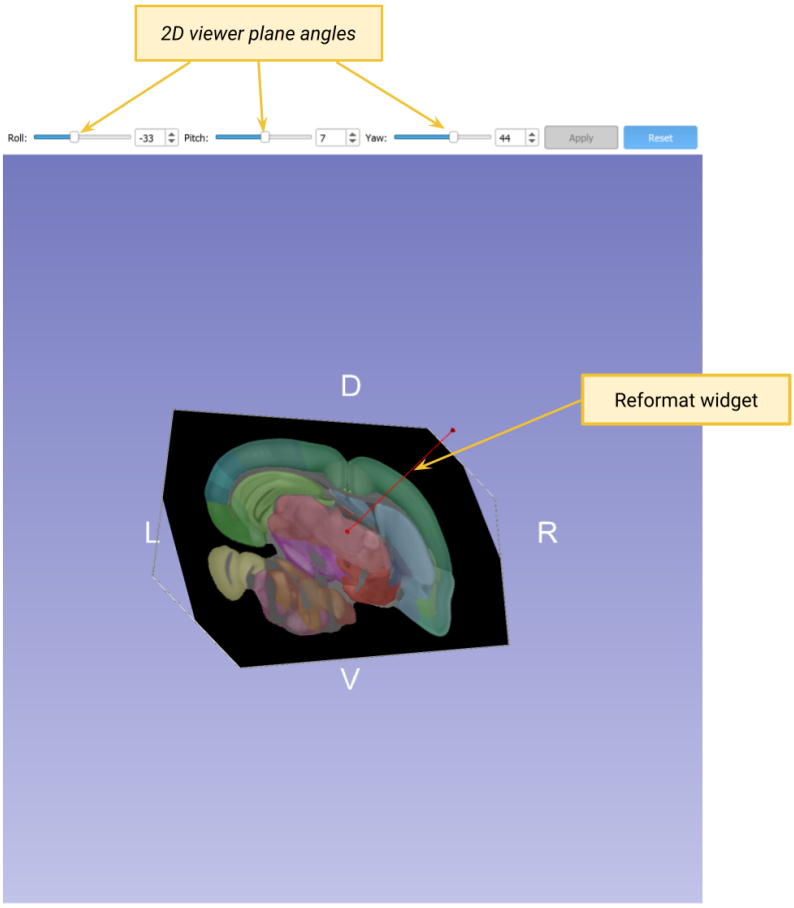
The 2D viewer (or slice viewer) allows to create or edit annotations on the viewing plane. The slice orientation allows to change the reference from which row, pitch and yaw angles are set in the 3D viewer.



- **Slice offset slider** allows slicing through the atlas reference volume. Step size is set in the *Property Editor*.
- **Adjust field of view** centers the slice viewer's field of view to match the extent of the atlas reference volume.
- **Slice orientation** allows to choose the orientation of the slice in the 2D and *3D Viewer*.
- **Contrast slider** allows to adjust the grayscale background contrast associated with the atlas reference volume.
- **Reset contrast** allows to reset the grayscale background contrast associated with the atlas reference volume

3D Viewer

The 3D viewer provides an interface to update the slice viewer plane orientation.



- **Raw**, **Pitch** and **Yaw** sliders and input boxes allow to set the *2D Viewer* plane angles relative to the selected slice orientation (Axial, Coronal or Sagittal).
- **Apply** allows to update the slice orientaton based on the set angles.
- **Reset** allows to reset the angles to match the slice orientation selected in the 2D Viewer.
- **Reformat widget** allows to update the slice orientation.

Property Editor

- **Slice Step Size** configures how the *2D Viewer* slicer offset may be adjusted.
- **Annotation Thickess** updates the width of selected `Curve` annotation.
- **Annotation Type** updates how the annotation is represented in both viewers.

Numeric Inputs

After clicking in either the **Slice Step Size** or **Annotation Thickess** spin box, its value may be updated using a "large" step as described in the *Numeric Inputs* keyboard shortcuts.

Status Bar

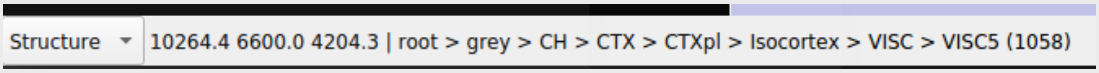
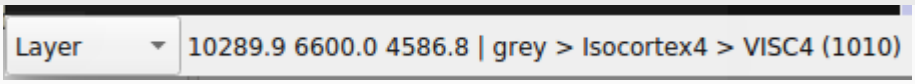
- **Ontology Selector** allows to select between `None`, `Structure` and `Layer` (only available for the CFF atlas).
- **Text** about what is visible at the current mouse pointer position.

The status bar text is formatted as

```
x y z | <path> (<label index>)
```

where:

- `x y z` corresponds to the world coordinates (RAS).
- <path> is formatted as `> structure 1 > structure 2 > ...` (or `> layer 1 > layer 2 > ...`).
- <`label index`> corresponds to the value in the annotation volume.

Ontology	Example of text
structure	
layer	

1.2.3 Keyboard and Mouse Shortcuts

On macOS use the Command key () instead of the Control (Ctrl) key

General

Key	Effect
`Ctrl + n`	Create a new annotation
`Ctrl + s`	Save current annotation(s)
`Ctrl + o`	Load annotation(s) from file
`Ctrl + w`	Reset views discarding current changes
`f`	Increment Slice offset
`b`	Decrement Slice offset
`r`	Adjust field of view to match the extent of the atlas

The user will be prompted with a save dialog before overwriting unsaved changes.

2D Viewer - Zoom and Pan

These interactions are enabled only in *Explore* and *Edit* mode.

Interface Device	Zoom	Pan
3-button mouse	Vertical drag `Right-Click`	Drag `Middle-Click`
2-button mouse	Vertical drag `Right-Click`	Drag `Shift + Left-Click`
1-button mouse	Vertical drag `Ctrl + Left-Click`	Drag `Shift + Left-Click`
Trackpad	Vertical drag two fingers	Drag `Shift + Left-Click`

2D Viewer - Annotation

These interactions are enabled only in *Edit* and *Place* modes.

Key or mouse operation	Effect
`Left-Click` on annotation line	Add annotation point
`Right-Click` on point then `Delete`	Delete currently selected annotation point
`Right-Click` on point then `Interaction`	Show/hide the interaction widget
`Ctrl + Left-Click` on annotation line	Insert annotation point
Drag `Alt + Left-Click` on annotation line	Rotate annotation
Drag `Alt + Right-Click` on annotation line	Scale annotation
Drag `Middle-Click` on annotation line	Translate annotation

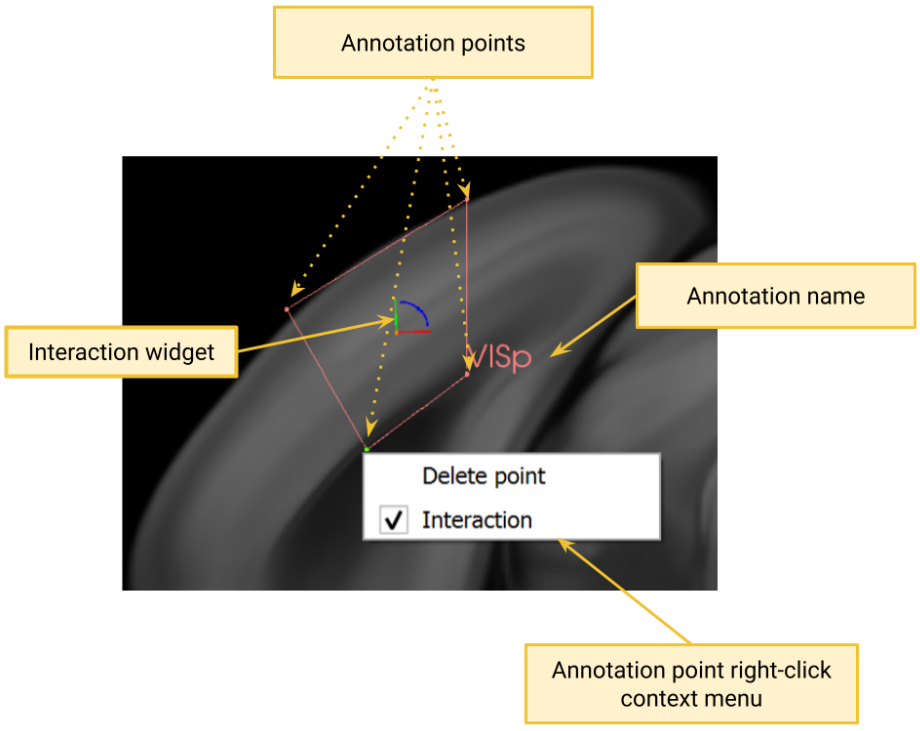


Fig. 3: Annotation point right-click context menu.

Numeric Inputs

`Left`/`Right` keys increment by 1, and `PgUp`/`PgDn` keys increment by a contextual "large" step.

Large step for angle inputs is 5° and for distance inputs is 10mm. The `Slice Offset` steps in multiples of the `Slice Step Size`.

Annotation List

Action	Effect
Double-`Left-Click` annotation	Edit annotation name
`Left-Click` on eye icon	Show/hide annotation

1.2.4 Command-line arguments

Cell Locator supports the same CLI arguments as 3D Slicer, as well as the following:

General options

`--annotation-file` `<path>`
: Path to an existing annotation file to be immediately loaded.

`--reference-view` `<view>`
: Initial slice position. View may be one of `Axial`, `Coronal`, `Saggital`. Default value is `Coronal`

`--view-angle` `<angle>`
: View angle in degrees; specifies an angle from –reference-view.

`--atlas-type` `<type>`
: The atlas type to load. Type may be one of `ccf`, `mni`. If not provided, prompt user before startup.

LIMS options

See *LIMS Integration* for LIMS API documentation.

`--lims-specimen-id` `<id>`
: LIMS specimen id to retrieve and load.

`--lims-specimen-kind` `<kind>`
: LIMS specimen kind to load or save. Default is `'IVSCC cell locations'`.

`--lims-base-url` `<url>`
: LIMS base url.

1.3 Reference Atlases

Cell Locator bundles the files associated with the reference atlases described below.

1.3.1 Allen Mouse Brain Common Coordinate Framework (CCF)

Version 3

Allen Mouse Brain Common Coordinate Framework (CCFv3) is a 3D reference space by creating an average brain at 10um voxel resolution from serial two-photon tomography images of 1,675 young adult C57Bl6/J mice.

Using multimodal reference data, the entire brain was directly parcellated in 3D, labeling every voxel with a brain structure spanning 43 isocortical areas and their layers, 314 subcortical gray matter structures, 81 fiber tracts, and 8 ventricular structures.

Allen Mouse Brain Common Coordinate Framework (CCFv3):

- Details: https://community.brain-map.org/t/allen-mouse-ccf-accessing-and-using-related-data-and-tools/359
- Research Resource Identifier (RRID): RRID:SCR_020999

Annotation Volume	
Download	http://download.alleninstitute.org/informatics-archive/current-release/mouse_ccf/annotation/ccf_2017/
Filename	`annotation_25.nrrd`
Checksum (SHA256)	`c620cbcc562183e4dcd40250d440130501781f74b41de35b1c1bdabace290c42`

Reference Volume	
Download	http://download.alleninstitute.org/informatics-archive/current-release/mouse_ccf/average_template/
Filename	`average_template_25.nrrd`
Checksum (SHA256)	`e4a2b483e842b4c8c1b5452d940ea59e14bc1ebaa38fe6a9c3bacac6db2a8f4b`

1.3.2 Allen Human Reference Atlas - 3D

Version 1.0.0 (2020)

The Allen Human Reference Atlas - 3D, is a parcellation of the adult human brain in 3D, labeling every voxel with a brain structure spanning 141 structures.

Annotation Volume	`Allen Human Reference Atlas - 3D, 2020`
Version	1.0.0
Details	https://community.brain-map.org/t/allen-human-reference-atlas-3d-2020-new/405
Research Resource Identifier (RRID)	RRID:SCR_017764
Download	http://download.alleninstitute.org/informatics-archive/allen_human_reference_atlas_3d_2020/version_1/
Filename	`mni_annotation_full.nii.gz`
Checksum (SHA256)	`2b05581e39c44f2623d9b0a69f64e3df0823c20d054abef92973812313335dc3`

These parcellations were drawn on the MRI reference brain volume "ICBM 2009b Nonlinear Symmetric", a non-linear average of the MNI152 database of 152 normal brain images.

Reference Volume	`ICBM 152 Nonlinear Symmetric`
Version	2009b
Details & Download	https://www.bic.mni.mcgill.ca/ServicesAtlases/ICBM152NLin2009
Filename	`mni_icbm152_t1_tal_nlin_sym_09b_hires.nii`
Checksum (SHA256)	`b2ca5ca7c94471f0ca77b64485c0d9655737b9029ec19ef7486ab89f1ce4bf79`

1.4 Coordinate Systems

1.4.1 Atlases

The table below describes the coordinate systems of the *reference atlases* distributed in Cell Locator.

Reference atlas	Atlas Version	Coordinate System Dimensions
Allen Mouse Brain Common Coordinate Framework	3	PIR
Allen Human Reference Atlas - 3D	1.0.0	RAS

By convention, the coordinate system dimensions are the the following:

- PIR (Posterior Inferior Right):
  - `X` axis is `Anterior-to-Posterior`
  - `Y` axis is `Superior-to-Inferior`
  - `Z` axis is `Left-to-Right`.
- RAS (Right Anterior Superior):
  - `X` axis is `Left-to-Right`
  - `Y` axis is `Posterior-to-Anterior`
  - `Z` axis is `Inferior-to-Superior`

1.4.2 Cell Locator

World space for 3D Views is in RAS (Right Anterior Superior) orientation where `x axis = Left-to-Right`, `y axis = Posterior-to-Anterior` and `z axis = Inferior-to-Superior`.

CCF Atlas integration

Since the orientation header in the `.nrrd` files used for the CCF are incorrect[1], Cell Locator workarounds the problem by applying two countermeasures:

1. It associates a `RAStoPIR` transform[2] with both the loaded CCF average template and annotation volumes..
2. It updates the slice orientation preset associated with the coronal viewer[3] so that `+x` is `+P`.

1.5 LIMS Integration

Integration with a Laboratory Information Management System (LIMS) is achieved through a simple REST api; there are only two endpoints to get and set annotations for a particular specimen.

Support is enabled by command line options *`--lims-specimen-id`* and *`--lims-base-url`*, and optionally *`--lims-specimen-kind`*.

- At application startup time to (1) retrieve the corresponding annotation from LIMS and load it by using the `<base-url>/specimen_metadata/view` endpoint and (2) to enable the "Upload Annotation" button
- After user initiate annotation upload to LIMS while using the `<base-url>/specimen_metadata/store` endpoint.

The `specimen_id` parameter for both requests is set to the value of *`--lims-specimen-id`*, and the `kind` parameter is set to the value of *`--lims-specimen-kind`*.

For testing the functionality a simple reference LIMS server is implemented using flask; this server is only suitable for testing the API. See Endpoint Documentation for more information and sample requests.

1.6 Command Line Tools

Utility CLI tools for processing Cell Locator annotation files.

For information on extending `cl-convert`, see the *Converter API Documentation*

[1] https://github.com/BICCN/cell-locator/issues/48
[2] https://github.com/BICCN/cell-locator/issues/48#issuecomment-443412860
[3] https://github.com/BICCN/cell-locator/issues/48#issuecomment-443423073

1.6.1 Installation

```
$ pip install cell-locator-cli
```

1.6.2 Tools

cl-export

Export Cell Locator annotations to VTK model or labelmap.

```
usage: cl-export [-h] [-m MODEL_PATH] [-l LABELMAP_PATH] [-a ATLAS_PATH]
                 [--pir]
                 annotation
```

annotation
: Input Cell Locator annotation file (JSON).

-h, --help
: show this help message and exit

-m <model_path>, **--model** <model_path>
: Output path for annotation model. If not provided, model generation is skipped.

-l <labelmap_path>, **--labelmap** <labelmap_path>
: Output path for annotation labelmap. If not provided, labelmap generation is skipped. Requires –atlas for spacing information.

-a <atlas_path>, **--atlas** <atlas_path>
: Atlas volume or labelmap. Used to set spacing/direction on the output labelmap.

--pir
: If set, read the annotation in PIR format rather than RAS. This should only be necessary for old-style CCF annotations.

cl-convert

A tool used to upgrade annotation .json files through breaking changes to the file format. The converter can also downgrade file formats, but this is not as thoroughly tested.

```
usage: cl-convert [-h] {convert,versions,infer} ...
```

-h, --help
: show this help message and exit

cl-convert convert

```
usage: cl-convert convert [-h] -v VERSION [-t TARGET] [--no-indent] src dst
```

src
: Source JSON file. Use '-' to read from stdin.

dst
: Destination JSON file. Use '-' to write to stdout.

-h, --help
: show this help message and exit

-v <version>, **--version** <version>
: Source file version. Use '-v?' to infer the version.

-t <target>, **--target** <target>
: Target file version. Defaults to the latest version.

--no-indent
: Do not indent output JSON.

cl-convert infer

```
usage: cl-convert infer [-h] src
```

src
: Source JSON file. Use '-' to read from stdin.

-h, --help
: show this help message and exit

cl-convert versions

```
usage: cl-convert versions [-h] [target]
```

target
: Show versions matching this target. If empty, show all versions.

-h, --help
: show this help message and exit

1.6.3 Sample Usage

Export an MNI annotation to labelmap and model

```
$ cl-export \
   mni-annotation.json \
  -m mni-annotation.vtk \
  -l mni-annotation.label.nrrd \
```

(continues on next page)

(continued from previous page)

```
  -a mni_annotation_contiguous.nrrd
$ f3d mni-annotation.vtk
$ f3d -v mni-annotation.label.nrrd
```

Export a CCF annotation to labelmap only

```
$ cl-export \
  ccf-annotation.json \
  -l ccf-annotation.label.nrrd \
  -a ccf_annotation_25_contiguous.nrrd \
  --pir
$ f3d ccf-annotation.vtk
$ f3d -v ccf-annotation.label.nrrd
```

Update an old annotation file

```
$ cl-convert convert -v'?' -t v0.2 older.json newer.json
Inferred version 'v0.0.0+2020.08.26'
$ cl-convert infer newer.json
v0.2.1+2022.03.04
```

1.6.4 Version Identifiers

cl_convert.converters.**match**(*target*)

Find the most-recent versions matching the target.

Prefix with `d` to interpret as a date. Prefix with `v`, or no prefix, to interpret as a literal version.

Example Versions	`v1.1`	`v1.1.`	`d2020.`
`1.1.0+2019.02.01`	yes	yes	no
`1.1.1+2020.02.07`	yes	yes	yes
`1.2.0+2020.05.01`	no	no	yes
`1.10.1+2021.03.01`	yes	no	no

Parameters

target – String describing target versions.

Returns

Matching versions, in order of precedence (most-recent first)

1.7 Developer Guide

1.7.1 Build and Package CellLocator

This document summarizes how to build and package CellLocator on Windows.

Cell Locator is a custom Slicer application. Reading the 3D Slicer Developer Documentation may help answer additional questions.

The initial source files were created using KitwareMedical/SlicerCustomAppTemplate.

Prerequisites

- Setting up your git account:
  - Create a Github account.
  - Setup your SSH keys following these instructions at the exception of `step 2` where you should **NOT** enter a passphrase.
  - Setup your git username and your git email.
  - If not already done, email `Lydia Ng <LydiaN@alleninstitute.org>` to be granted access to the BICCN/cell-locator repository.

Checkout

1. Start Git Bash
2. Checkout the source code into a directory `C:\W\` by typing the following commands:

```
cd /c
mkdir W
cd /c/W
git clone https://github.com/BICCN/cell-locator.git CL
```

Note: use short source and build directory names to avoid the maximum path length limitation.

Build

Note: The build process will take approximately 3 hours.

Option 1: CMake GUI and Visual Studio (Recommended)

1. Start CMake GUI, select source directory `C:\W\CL` and set build directory to `C:\W\CLR`.
2. Add an entry `Qt5_DIR` pointing to `C:/Qt/${QT_VERSION}/${COMPILER}/lib/cmake/Qt5`.
3. Generate the project.
4. Open `C:\W\CLR\CellLocator.sln`, select `Release` and build the project.

Option 2: Command Line

1. Start the Command Line Prompt
2. Configure and build the project in `C:\W\CLR` by typing the following commands:

```
cd C:\W\
mkdir CLR
cd CLR
cmake -G "Visual Studio 16 2019" -A x64 -DQt5_DIR:PATH=`C:/Qt/${QT_VERSION}/${COMPILER}/
↪lib/cmake/Qt5 ..\CL
cmake --build . --config Release
```

Package

Install NSIS 2

Option 1: CMake and Visual Studio (Recommended)

1. In the `C:\W\CLR\Slicer-build` directory, open `Slicer.sln` and build the `PACKAGE` target

Option 2: Command Line

1. Start the Command Line Prompt
2. Build the `PACKAGE` target by typing the following commands:

```
cd C:\W\CLR\Slicer-build
cmake --build . --config Release --target PACKAGE
```

1.7.2 Cell Locator API

Markups, Models, and Annotations

Each annotation in the scene has a markup associated with it, and also a translucent 3d model to represent the thickness of that annotation. So for each element in the scene, we need to manage a `vtkMRMLMarkupsNode` *and* a `vtkMRMLModelNode`. The `Annotation` class in Python does this and handles serialization to JSON.

`Annotation` is an abstract class, currently with two concrete implementations: `FiducialAnnotation` and `ClosedCurveAnnotation`.

`FiducialAnnotation` consists only of points with no thickness. Since this behavior is provided by `vtkMRMLMarkupsFiducialNode`; so the annotation does not need to update a model.

`ClosedCurveAnnotation` does have thickness. Its markup is a `vtkMRMLMarkupsClosedCurveNode`, and a model is generated from this markup using `vtkSlicerSplinesLogic::CreateModelFromContour` (see here)

Creating Annotation Types

Below is a template for creating a new type of Annotation. Keep in mind that `Annotation` does not create a model, so if you need one (or any other nodes) for the new type, you must create those in `NewAnnotation.__init__`.

```
class NewAnnotation(Annotation):
    DisplayName = 'Sample'   # default name for this annotation type
    MarkupType = 'vtkMRMLMarkupsFiducialNode'  # markup type managed by this annotation␣
↪type

    def __init__(self, markup=None):
        # set any type-specific attributes, models, etc

        super().__init__(markup=markup)

        # do any DisplayNode customizations

    def clear(self):
        # do any cleanup of type-specific attributes

    def update(self):
```

(continues on next page)

(continued from previous page)

```
        # update any type-specific attributes; models, etc

    def metadata(self):
        # convert type-specific attributes to dict for serialization

    def setMetadata(self, data):
        # set any type-specific attributes from dict for deserialization
```

(More info here)

`Annotation.update` is invoked each time the markup changes, so we can implement it to re-generate a model. Any other nodes or data that need to be synchronized with the markup can be updated here.

Serialization

Annotations are serialized to JSON using `Annotation.toDict`. The markup is converted to JSON by Slicer (see here). Any annotation-specific metadata (thickness, etc) is included using `Annotation.getMetdata`.

Deserialization

Annotations are deserialized using `Annotation.fromDict`. The specific annotation type is determined using the markup type, and the annotation type's `MarkupType` class attribute.

```
class PlaneAnnotation(Annotation):
    MarkupType = 'vtkMRMLMarkupsPlaneNode'
```

For example, if the above class is defined and a `vtkMRMLMarkupsPlaneNode` is stored in the JSON, then the resulting annotation will be an instance of `PlaneAnnotation`.

The JSON contents will then be sent to `PlaneAnnotation.setMetadata` so that any attributes specific to `PlaneAnnotation` may be deserialized.

1.7.3 cl-convert Converter API

Converter API

class cl_convert.model.**Converter**

Utility to aid in conversion of Cell Locator files.

"Specialized" – a dict representation of a version-specific JSON. That representation may only work in one version of Cell Locator.

"Normalized" – a dataclass representation common to all versions of cell locator. An intermediate representation during the conversion process.

For example, the flow to update a file to a different version would be:

```
>>> doc = old_converter.normalize(data)
>>> new_data = new_converter.specialize(doc)
```

It is also easier to perform manipulations on a `Document`. For example:

```
>>> doc = converter.normalize(data)
>>> for annotation in doc.annotations:
...     annotation.name = annotation.name.lower()
>>> data = converter.specialize(data)
```

abstract classmethod normalize(*data: dict*)

Convert a specialized dict to a normalized Document

abstract classmethod specialize(*doc:* Document)

Convert a normalized Document to a specialized dict; specific to this version.

`@cl_convert.model.`**versioned**

Automatically infer version string from calling filename.

When the caller's filename is formatted `'v{version}.py'`, extract `{version}` from that filename and set the `"version"` key in the result.

For example, in the file `v1.0.0.py`:

```
@versioned
def specialize():
    return {}

data = specialize()
assert data['version'] == '1.0.0'
```

Version Registration

`cl_convert.converters.`**find_latest**(*target: str = ''*) → Tuple[str, *Converter*]

Find the most-recent matching version and converter.

Parameters
target – The target version string. See match() for details on matching logic.

Returns
The inferred version and the corresponding converter.

`cl_convert.converters.`**infer_normalize**(*data: dict*) → Tuple[str, *Document*]

Find the most-recent converter that can normalize the document.

Returns
(version, document) — The inferred version and the normalized document.

`cl_convert.converters.`**match**(*target: str = ''*) → Generator[str, None, None]

Find the most-recent versions matching the target.

Prefix with `d` to interpret as a date. Prefix with `v`, or no prefix, to interpret as a literal version.

Example Versions	`v1.1`	`v1.1.`	`d2020.`
`1.1.0+2019.02.01`	yes	yes	no
`1.1.1+2020.02.07`	yes	yes	yes
`1.2.0+2020.05.01`	no	no	yes
`1.10.1+2021.03.01`	yes	no	no

Parameters
target – String describing target versions.

Returns
Matching versions, in order of precedence (most-recent first)

Document Model

class cl_convert.model.**Document**(*annotations: ~typing.List[~cl_convert.model.Annotation] = <factory>*, *current_id: int = 0*, *reference_view: str = 'Coronal'*, *ontology: str = 'Structure'*, *stepSize: float = 0.5*, *camera_position: ~typing.Tuple[float, float, float] = (51.6226, -631.3969, -605.9925)*, *camera_view_up: ~typing.Tuple[float, float, float] = (-0.5686, -0.6042, 0.5582)*)

Store minimal information about an annotation.json document.

camera_position: Tuple[float, float, float] = (51.6226, -631.3969, -605.9925)
Initial camera position.

camera_view_up: Tuple[float, float, float] = (-0.5686, -0.6042, 0.5582)
Initial camera 'up' vector.

current_id: int = 0
Index of the currently-selected annotation.

ontology: str = 'Structure'
Initial atlas ontology. Ex. 'Structure', 'Layer', or 'None'

reference_view: str = 'Coronal'
Initial reference view. Ex. 'Coronal', 'Axial', or 'Sagittal'.

stepSize: float = 0.5
Distance in *Annotation.coordinate_units* to move slice plane in Explore mode.

class cl_convert.model.**Annotation**(*name: str = ''*, *markup_type: str = 'ClosedCurve'*, *representation_type: str = 'spline'*, *thickness: float = 50*, *coordinate_system: str = 'LPS'*, *coordinate_units: str = 'um'*, *orientation: ~typing.Tuple[float, float, float, float, float, float, float, float, float, float, float, float, float, float, float, float] = (1.0, 0.0, 0.0, 0.25, 0.0, 0.0, 1.0, -17.5, 0.0, 1.0, 0.0, 22.25, 0.0, 0.0, 0.0, 1.0)*, *points: ~typing.List[~cl_convert.model.Point] = <factory>*)

Store minimal information about a single annotation

coordinate_system: str = 'LPS'
Should always be LPS here; older versions of Slicer use RAS.

coordinate_units: str = 'um'
Should be um for CCF atlas, mm for MNI atlas.

orientation: Tuple[float, float, float, float, float, float, float, float, float, float, float, float, float, float, float, float] = (1.0, 0.0, 0.0, 0.25, 0.0, 0.0, 1.0, -17.5, 0.0, 1.0, 0.0, 22.25, 0.0, 0.0, 0.0, 1.0)
A transformation matrix storing the orientation of the slicing plane.

points: List[Point]
Control point positions for the annotation markup.

representation_type: str = 'spline'

Type for a closed curve annotation; ex 'spline' or 'polyline'.

thickness: float = 50

Thickness of the annotation model

1.7.4 Annotation File Format

> **Warning:** The Cell Locator annotation format is non-standard and exists purely as an implementation detail. While we are documenting *structural changes* and providing a *converter*, the file organization may change from version to version without notice.
>
> We mean it.

Modifying the File Format

When a change is made to the file format (e.g. a new type of Annotation is added), be sure to update the conversion script and documentation

- Follow semantic versioning to increment `AnnotationManager.FORMAT_VERSION` in `Home.py`. Use the date of the release as the build metadata.
- Update the conversion script
  - Create a converter in versions. It's easiest to copy the most-recent converter and modify the `specialize` and `normalize` methods accordingly. See the *Converter API*
  - Update `version_order` in converters.py. Add the new version number at the top of the list; this way `latest_version` will point to the new converter.
- Update the *version history* below.

Versioning Guidelines

This section provides hints for updating the `AnnotationManager.FORMAT_VERSION` constant set in Modules/Scripted/Home/Home.py.

Based on the answer to the following question:

> Since the last release of Cell Locator, were annotation format changes introduced ?

Changes are usually introduced following the Modifying the File Format *instructions.*

If the answer is **No**, consider creating a *synchronization* converter script:

1. copy the most recent `vX.Y.Z+YYYY.MM.DD.py` script. This will ensure the annotation saved with the new release of cell locator will have a version string newer than the last release.
2. add an entry to the `version_order` list in converters.py script.
3. add an entry in the *Versions* section below.
4. add an entry in the [version-changlist.md][version-changlist] document.

If the answer is **yes**, no changes are required.

Versions

The latest version listed below should correspond to the version hard-coded in the `AnnotationManager.FORMAT_VERSION` *constant set in Modules/Scripted/Home/Home.py.*

Overview of differences between versions are documented in the version changelist.

0.2.1 (2022-03-04)

Add the `structure` object to control point. `null` for values with missing structure (outside of atlas, converted from old file, etc.)

0.2.0 (2021-08-12)

Unchanged from 0.1.1; this version synchronizes the file format and cell locator release.

0.1.1 (2021-06-11)

Removed the `"measurements"` key from markups.


```
{
    "version": "0.1.1+2021.06.11",
    "markups": [
        {
            "markup": {
                "type": "ClosedCurve",
                "coordinateSystem": "LPS",
                "controlPoints": [
                    {
                        "id": "1",
                        "position": [
                            -7725.476777936878,
                            5071.924649397559,
                            -2858.6838622146615
                        ],
                        "orientation": [
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            1.0
                        ]
                    },
                    {
                        "id": "2",
                        "position": [
                            -8785.61316343005,
```


(continues on next page)

(continued from previous page)

```
                    5514.035987881592,
                    -4251.158410257692
                ],
                "orientation": [
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
            },
            {
                "id": "3",
                "position": [
                    -7732.355384361564,
                    5984.77667260136,
                    -4339.759187924226
                ],
                "orientation": [
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
            },
            {
                "id": "4",
                "position": [
                    -6980.525470132225,
                    5682.868336976309,
                    -3371.0529631232225
                ],
                "orientation": [
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
```

(continues on next page)

(continued from previous page)


```
                    },
                    {
                        "id": "5",
                        "position": [
                            -6735.759199784232,
                            5461.593738502808,
                            -2856.7326570107934
                        ],
                        "orientation": [
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            1.0
                        ]
                    }
                ]
            },
            "name": "Annotation",
            "orientation": [
                0.9423723483536421,
                -0.10260749019479301,
                -0.3184431817677528,
                5686.499999999999,
                0.32619099971630533,
                0.49341126738558755,
                0.8063155417831328,
                -6574.999999999999,
                -0.07438943985890348,
                0.863722770437872,
                -0.4984677455532305,
                -3987.4999999999995,
                0.0,
                0.0,
                0.0,
                1.0
            ],
            "representationType": "spline",
            "thickness": 50
        }
    ],
    "currentId": 0,
    "referenceView": "Coronal",
    "ontology": "Structure",
    "stepSize": 24.999999999999996,
    "cameraPosition": [
        5864.945206940424,
        -50233.77100882346,
```


(continues on next page)

(continued from previous page)

```
        -4015.7997990419617
    ],
    "cameraViewUp": [
        0.0,
        0.0,
        1.0
    ]
}
```

0.1.0 (2020-09-18), Unversioned (2020-09-18)

Annotation files generated with Cell-Locator 0.1.0-2020-09-18 do not include the `version` key.

```
{
    "version": "0.1.0+2020.09.18",
    "markups": [
        {
            "markup": {
                "type": "ClosedCurve",
                "coordinateSystem": "LPS",
                "measurements": [],
                "controlPoints": [
                    {
                        "id": "1",
                        "position": [
                            -7725.476777936878,
                            5071.924649397559,
                            -2858.6838622146615
                        ],
                        "orientation": [
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            -1.0,
                            0.0,
                            0.0,
                            0.0,
                            1.0
                        ]
                    },
                    {
                        "id": "2",
                        "position": [
                            -8785.61316343005,
                            5514.035987881592,
                            -4251.158410257692
                        ],
                        "orientation": [
                            -1.0,
                            0.0,
```

(continues on next page)

(continued from previous page)

```
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
            },
            {
                "id": "3",
                "position": [
                    -7732.355384361564,
                    5984.77667260136,
                    -4339.759187924226
                ],
                "orientation": [
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
            },
            {
                "id": "4",
                "position": [
                    -6980.525470132225,
                    5682.868336976309,
                    -3371.0529631232225
                ],
                "orientation": [
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    -1.0,
                    0.0,
                    0.0,
                    0.0,
                    1.0
                ]
            },
            {
                "id": "5",
                "position": [
                    -6735.759199784232,
                    5461.593738502808,
```

(continues on next page)

(continued from previous page)


```
                        -2856.7326570107934
                    ],
                    "orientation": [
                        -1.0,
                        0.0,
                        0.0,
                        0.0,
                        -1.0,
                        0.0,
                        0.0,
                        0.0,
                        1.0
                    ]
                }
            ]
        },
        "name": "Annotation",
        "orientation": [
            0.9423723483536421,
            -0.10260749019479301,
            -0.3184431817677528,
            5686.499999999999,
            0.32619099971630533,
            0.49341126738558755,
            0.8063155417831328,
            -6574.999999999999,
            -0.0743894398589034 8,
            0.863722770437872,
            -0.49844677455532305,
            -3987.4999999999995,
            0.0,
            0.0,
            0.0,
            1.0
        ],
        "representationType": "spline",
        "thickness": 50
    }
],
"currentId": 0,
"referenceView": "Coronal",
"ontology": "Structure",
"stepSize": 24.999999999999996,
"cameraPosition": [
    5864.945206940424,
    -50233.77100882346,
    -4015.7997990419617
],
"cameraViewUp": [
    0.0,
    0.0,
    1.0
```


(continues on next page)

(continued from previous page)

```
    ]
}
```

Unversioned (2020-08-26)


```
{
    "markups": [
        {
            "markup": {
                "type": "ClosedCurve",
                "coordinateSystem": "LPS",
                "locked": false,
                "labelFormat": "%N-%d",
                "controlPoints": [
                    {
                        "id": "1",
                        "label": "MarkupsClosedCurve-1",
                        "description": "",
                        "associatedNodeID": "vtkMRMLScalarVolumeNode1",
                        "position": [
                            -7725.476777936878,
                            5071.924649397559,
                            -2858.6838622146615
                        ],
                        "orientation": [
                            -1,
                            0,
                            0,
                            0,
                            -1,
                            0,
                            0,
                            0,
                            1
                        ],
                        "selected": true,
                        "locked": false,
                        "visibility": true,
                        "positionStatus": "defined"
                    },
                    {
                        "id": "2",
                        "label": "MarkupsClosedCurve-2",
                        "description": "",
                        "associatedNodeID": "vtkMRMLScalarVolumeNode1",
                        "position": [
                            -8785.61316343005,
                            5514.035987881592,
                            -4251.158410257692
                        ],
```


(continues on next page)

(continued from previous page)


```
                "orientation": [
                    -1,
                    0,
                    0,
                    0,
                    -1,
                    0,
                    0,
                    0,
                    1
                ],
                "selected": true,
                "locked": false,
                "visibility": true,
                "positionStatus": "defined"
            },
            {
                "id": "3",
                "label": "MarkupsClosedCurve-3",
                "description": "",
                "associatedNodeID": "vtkMRMLScalarVolumeNode1",
                "position": [
                    -7732.355384361564,
                    5984.776672600136,
                    -4339.759187924226
                ],
                "orientation": [
                    -1,
                    0,
                    0,
                    0,
                    -1,
                    0,
                    0,
                    0,
                    1
                ],
                "selected": true,
                "locked": false,
                "visibility": true,
                "positionStatus": "defined"
            },
            {
                "id": "4",
                "label": "MarkupsClosedCurve-4",
                "description": "",
                "associatedNodeID": "vtkMRMLScalarVolumeNode1",
                "position": [
                    -6980.525470132225,
                    5682.868336976309,
                    -3371.0529631232225
                ],
```


(continues on next page)

(continued from previous page)


```
                    "orientation": [
                        -1,
                        0,
                        0,
                        0,
                        -1,
                        0,
                        0,
                        0,
                        1
                    ],
                    "selected": true,
                    "locked": false,
                    "visibility": true,
                    "positionStatus": "defined"
                },
                {
                    "id": "5",
                    "label": "MarkupsClosedCurve-5",
                    "description": "",
                    "associatedNodeID": "vtkMRMLScalarVolumeNode1",
                    "position": [
                        -6735.759199784232,
                        5461.593738502808,
                        -2856.7326570107934
                    ],
                    "orientation": [
                        -1,
                        0,
                        0,
                        0,
                        -1,
                        0,
                        0,
                        0,
                        1
                    ],
                    "selected": true,
                    "locked": false,
                    "visibility": true,
                    "positionStatus": "defined"
                }
            ],
            "display": {
                "visibility": true,
                "opacity": 1,
                "color": [
                    0.4,
                    1,
                    1
                ],
                "selectedColor": [
```


(continues on next page)

(continued from previous page)


```
                    1,
                    0.5000076295109483,
                    0.5000076295109483
                ],
                "propertiesLabelVisibility": true,
                "pointLabelsVisibility": false,
                "textScale": 3,
                "glyphType": "Sphere3D",
                "glyphScale": 1,
                "glyphSize": 5,
                "useGlyphScale": true,
                "sliceProjection": false,
                "sliceProjectionUseFiducialColor": true,
                "sliceProjectionOutlinedBehindSlicePlane": false,
                "sliceProjectionColor": [
                    1,
                    1,
                    1
                ],
                "sliceProjectionOpacity": 0.6,
                "lineThickness": 0.2,
                "lineColorFadingStart": 1,
                "lineColorFadingEnd": 10,
                "lineColorFadingSaturation": 1,
                "lineColorFadingHueOffset": 0,
                "handlesInteractive": false,
                "snapMode": "toVisibleSurface"
            }
        },
        "orientation": [
            0.9423723483536421,
            -0.10260749019479301,
            -0.3184431817677528,
            5686.499999999999,
            0.32619099971630533,
            0.49341126738558755,
            0.8063155417831328,
            -6574.999999999999,
            -0.07438943985890348,
            0.863722770437872,
            -0.49844677455532305,
            -3987.4999999999995,
            0,
            0,
            0,
            1
        ],
        "representationType": "spline",
        "thickness": 50
    }
],
"currentId": 0,
```


(continues on next page)

(continued from previous page)

```
    "referenceView": "Coronal",
    "ontology": "Structure",
    "stepSize": 24.999999999999996,
    "cameraPosition": [
        5864.945206940424,
        -50233.77100882346,
        -4015.7997990419617
    ],
    "cameraViewUp": [
        0,
        0,
        1
    ]
}
```

Unversioned (2019-01-26)

```
{
    "DefaultCameraPosition":[
       5873.05421548901,
       -54260.69374827002,
       -4027.878783549626
    ],
    "DefaultCameraViewUp":[
       0.0,
       0.0,
       1.0
    ],
    "DefaultOntology":"Structure",
    "DefaultReferenceView":"Coronal",
    "DefaultRepresentationType":"spline",
    "DefaultSplineOrientation":[
       -1.0,
       0.0,
       0.0,
       5686.499999999999,
       0.0,
       0.0,
       -1.0,
       -6599.999999999999,
       0.0,
       1.0,
       0.0,
       -3987.4999999999997,
       0.0,
       0.0,
       0.0,
       1.0
    ],
    "DefaultStepSize":24.999999999999998,
```

(continues on next page)

(continued from previous page)


```
    "DefaultThickness":50.0,
    "Locked":0,
    "MarkupLabelFormat":"%N-%d",
    "Markups":[
       {
          "AssociatedNodeID": "vtkMRMLModelNode5",
          "CameraPosition": [
              0.0,
              0.0,
              0.0
          ],
          "CameraViewUp": [
              0.0,
              0.0,
              0.0
          ],
          "Closed": 1,
          "Description": "",
          "ID": "vtkMRMLMarkupsSplinesNode_0",
          "Label": "Annotation-1",
          "Locked": 0,
          "Ontology": "Structure",
          "OrientationWXYZ": [
              0.0,
              0.0,
              0.0,
              1.0
          ],
          "Points": [
              {
                  "x": 7736.406932105033,
                  "y": -6315.297589137253,
                  "z": -1755.7577043808915
              },
              {
                  "x": 8089.826952763821,
                  "y": -6315.297589137253,
                  "z": -1893.3464794555367
              },
              {
                  "x": 8563.263519088907,
                  "y": -6315.297589137253,
                  "z": -950.1244332210372
              },
              {
                  "x": 8233.71868467818,
                  "y": -6315.297589137253,
                  "z": -786.6443712232194
              }
          ],
          "Points_Count": "4",
          "ReferenceView": "Coronal",
```


(continues on next page)

(continued from previous page)


```
            "RepresentationType": "spline",
            "Selected": 1,
            "SplineOrientation": [
                -1.0,
                0.0,
                0.0,
                5658.88223569956,
                0.0,
                0.0,
                -1.0,
                -6315.297589137253,
                0.0,
                1.0,
                0.0,
                -2943.1556456320488,
                0.0,
                0.0,
                0.0,
                1.0
            ],
            "StepSize": 735.0,
            "Thickness": 50.0,
            "Visibility": 1
        }
    ],
    "Markups_Count":0,
    "TextList":[
       null
    ],
    "TextList_Count":0
}
```


1.7.5 Annotation File Version Changlist

Base

```
+ Locked
+ MarkupLabelFormat
+ Markups []
+ Markups_Count
+ TextList [null]
+ TextList_Count 0
```

2019-01-26 -> 2020-04-16

2020-04-16, 2020-04-30, and 2020-08-26 all use the same format.

```
+ DefaultCameraPosition
+ DefaultCameraViewUp
+ DefaultOntology
+ DefaultReferenceView
+ DefaultRepresentationType
+ DefaultSplineOrientation
+ DefaultStepSize
+ DefaultThickness
```

2020-04-16 -> 2020-08-26

Major Change

```
! DefaultCameraPosition      -> cameraPosition
! DefaultCameraViewUp        -> cameraViewUp
! DefaultOntology            -> ontology
! DefaultReferenceView       -> referenceView
! DefaultStepSize            -> stepSize

- DefaultRepresentationType
- DefaultSplineOrientation
- DefaultThickness

- Locked
! MarkupLabelFormat -> markups[].markup.labelFormat
- Markups_Count
- TextList
- TextList_Count

! Markups[] (each element):
!     OrientationWXYZ -> orientation
!     RepresentationType -> representationType
!     Thickness -> thickness

+     markup
+         type
+         coordinateSystem
+         locked
+         labelFormat
+         controlPoints []
+             id
+             label
+             description
+             associatedNodeID
+             position []
+             orientation []
+             selected
+             locked
```

(continues on next page)

(continued from previous page)

```
+             visibility
+             positionStatus
+         display
+             <displaynode>

-     AssociatedNodeID
-     CameraPosition
-     CameraViewUp
-     Closed
-     Description
-     ID
-     Label
!     Locked -> markup.locked
-     Ontology
!     Points[][] -> markup.controlPoints[].position[]
-     Poinst_Count
-     ReferenceView
-     Selected
-     StepSize
-     Visibility
```

2020-08-26 -> 2020-09-18

```
! markups[]
+     name
!     markup
-         locked
-         labelFormat
-         display
!         controlPoints
-             associatedNodeID
-             description
-             label
-             locked
-             positionStatus
-             selected
-             visibility
```

2020-09-18 -> 2021-06-08

```
+ version
! markups[]
+     coordinateUnits
+     measurements []
+         name
+         enabled
+         printFormat
+         units?
```

2021-06-08 -> 2021-06-11

```
! markups[]
-     measurements
```

2021-06-11 -> 2021-08-12

Unchanged; this version synchronizes the file format and cell locator release.

2021-08-12 -> 2022-03-04

```
! markups[]
!     markup
!         controlPoints
+             structure
```

1.7.6 Release Notes

Next Release

Documentation:

- Explicitly reference license in project `README` and generated documentation main page. #227
- Fix documentation generation adding `.readthedocs.yaml` to explicitly desxcribe the documentation build environment. #227

Cell Locator 0.3.0-2022-07-25

Features:

- Add structure ID and acronym to JSON format. #208
- Add optional `lims-specimen-kind` command line parameter. #213

Fixes:

- Don't show exit confirmation when file (or LIMS specimen) is unchanged. #204

Documentation:

- Document double-click action to rename annotations. #206
- Update expected Slice Offset values in QA template. #203; see #139 (comment) for details.

Cell Locator 0.2.0 2021-08-12

Features:

- Created script to convert between Cell Locator file formats. #181, #182

Fixes:

- Fixed new `"measurements"` key being incorrectly included in serialization. #182
- Fixed changing ontology resetting slice view. #178
- Fixed GUI desync when editing multiple annotations. #179, #180
- Fixed polyline annotations incorrectly loading as splines. #184
- Fixed point annotations persisting after scene clear. #185

Documentation:

- Document the Annotation type hierarchy.
- Provide documentation and example of how to create new Annotation types.
- Document the mock LIMS server and LIMS API.
- Switch from recommonmark to MyST; so that reStructuredText may be avoided entirely. See Slicer#5662.
- File Format Version bumped to `0.2.0+2021-08-12` #198

Cell Locator 0.1.0 2021-05-21

Features:

- Add support for adding "point" annotations. See #164 and #164.
- Add support for user selection of atlas through the UI at startup if no atlas is specified on the command line. See #165
- Add a version number to the file format using semantic versioning. See #170

Fixes:

- Fix orientation of MNI atlas. See #163
- Use full annotations for MNI atlas, instead of single side annotation. See #164
- Fix errors when changing interaction mode while there are no annotations in the scene. See #173

Documentation:

- Add Documentation/CoordinateSystem.md with `Updates` section.
- Add versioning history to Documentation/AnnotationFileFormat.md. The current version is `0.1.0+2020.09.18`

Cell Locator 0.1.0 2020-09-18

Features:

- Add support for editing multiple annotations in one file. See #90
  - List of annotations displayed on left sidebar.
  - Add button for *adding*, *cloning*, and *removing?* annotations. See #102
- Responsive UI depending on whether CellLocator was launched by LIMS
  - If launched with LIMS enabled, hide the file controls (New, Save, Load, Save As). Otherwise, hide the LIMS controls (Save to LIMS)
- Save and load annotations by name
  - Double-click annotation to rename
- Display interaction handles allowing to rotate or translate current annotation. See #141 and #118
- Display error dialog when LIMS connection failed to be established.
- Add support for loading MNI atlas. See #99
  - Accept `--atlas-type` command line argument.
  - Supported value for atlas type are `mni` and `ccf`.
  - Display `mm` unit suffix for `mni` atlas and `um` for `ccf` atlas.

File format:

- Simplify annotation format removing unneeded keys. See #150

Fixes:

- Drawing a second polygon results in an empty json file. See #108
- Use of contrast "reset" button. See #136
- Use of Ctrl+W shortcut to reset application state. See #134
- Crash on application shutdown. See #154
- Remember last saved directory. See #137
- Initialize annotation base name to `Annotation` instead of `MarkupsClosedCurve`.
- Ensure only current annotation can be updated when `annotate` or `place` mode is activated.
- Hide irrelevant annotation control point right click menu entries.
- Handle annotation payload with missing `data` key.
- Fix annotation scaling. See #156
- Spline polygons do not save, but also do not disappear in a session involving multiple files. See #112
- After using Ctrl+w and restarting the application, `mm` unit prefix is displayed instead of `um`. See #89
- Support loading annotation from network path. See #103

Documentation:

- Add MAINTAINERS.md with `Making a release` section.
- Update `Keyboard Accelerators and Mouse Operations / Annotation` section in `README.md`.

Testing:

- Update AllenInstituteMockLIMS adding *Getting Started* section to `README.md`.

Cell Locator 0.1.0 2020-08-26

Features:

- In addition of `Explore` and `Annotate` mode, introduce the `Place` interaction mode.
  - `Annotate`: Used for editing existing control points.
  - `Place`: Used for adding new control points
  - `Explore`: Used for exploring the space

Infrastructure:

- Update Slicer to version from 2020-08-14. See KitwareMedical/Slicer@cell-locator-v4.11.0-2020-08-14-376d405c2b and #96
- Update to Python 3
- Introduce `Annotation` and `AnnotationManager` classes for interfacing with built-in Slicer markups nodes.

Fixes:

- Improve LIMS integration to address the feedback in #93.
  - Switch to the `requests` library to properly handle request headers. POST requests use the `json` argument, which sets `Content-Type: application/json`.
  - Change failure messages to be more helpful. Now includes the error status code and message. For example: `Failed to load annotations for LIMS specimen <ID>. Error <STATUS>: '<REASON>'`
  - Current LIMS specimen ID is shown in the file path box below "Save to LIMS".
- Ensure updating annotation point do not move the whole polygon. See #109
- Improve insertion of point in existing annotation. See #80

Cell Locator 0.1.0 2020-07-30

Features:

- Add LIMS support. See #93
  - Support command-line flags `--lims-specimen-id` and `--lims-base-url`.
  - Add new section to README

Tests:

- Add mock server to test LIMS functionalities. See instructions at https://github.com/KitwareMedical/AllenInstituteMockLIMS

Cell Locator 0.1.0 2020-04-30

Features:

- Add support for `--reference-view`, `--view-angle` and `--annotation-file` command-line arguments. See #97

Fixes:

- Ensure that +x is +P in both slice view and 3D view. See #101
- Ensure load dialog is associated with directory last used for loading. See #105

Cell Locator 0.1.0 2020-04-16

Fixes:

- Ensure Coronal referenceView is coherent between 2D and 3D view. See #101

Cell Locator 0.1.0 2019-06-03

Features:

- Reset camera position on reset. See #71
- Leave the camera alone when creating a new annotation. See #70 Motivation: better workflow for pinning multiple cells to the same slice
- Add a window/level slider for the background image. See #69
- Enable 2D Viewer pan and zoom. See #68
- Support for loading an annotation using `Ctrl`+`O` shortcut
- Support saving of view properties even if no annotation was added

Documentation:

- Add "Known Issues" section to README
- Add issue templates for QA, feature request and bug report
- Update "Keyboard Accelerators and Mouse Operations"

Fixes:

- Ensure referenceView is not reset when adding point to an annotation. See #72
- Ensure entering Roll/Pitch/Yaw values does NOT automatically Apply. See #75 and #67
- Ensure Sagittal referenceView is coherent between 2D and 3D view. See #74
- Fix handling of New/Save/SaveAs/Load. See #65
- Fix "Load -> Cancel -> SaveAs" workflow. See #86
- Disable unneeded keyboard shortcuts in 3D viewer
- Update roll/pitch/yaw slides to use single step of 1 and pageStep of 5. See #82
- Only associate Unit prefix with StepSize and Thickness sliders
- Ensure loaded annotation are snapped when switching to edit mode
- Ensure load dialog is associated with directory last use for saving. See #85

Cell Locator 0.1.0 2019-02-19

Documentation:

- Add "Keyboard Accelerators" section to README

Fixes:

- Rename S/I axis to D/V. See #66
- Fix transparency of application windows icon. See #66
- Set dialog title when creating new annotation. See #66
- Improve logo. See #13
- Ensure launcher splashscreen remains visible

Cell Locator 0.1.0 2019-02-15

Fixes:

- Improve handling of default save location. By default, the "Documents" location is used. In subsequent save operation, the last known saved directory is suggested. See #62
- Do not keep track of last annotation save directory if unsuccessful saving. See #62
- Re-order orientation sliders
- Ensure the "Apply" button is always enabled when setting roll/pitch/raw
- Ensure slice orientation is restored when loading annotation
- Update SplashScreen, logo and icon. See #13

Cell Locator 0.1.0 2019-01-26

Features:

- Add support for changing annotation type from "spline" to "polyline". #57
- Support selecting "None" Color. See #61
- Pretty-print serialized annotation file. See #63
- Add reset field of view button. See #64
- Serialize ReferenceView, StepSize, Ontology and Camera Position&ViewUp in annotation json file. See #22
- Add `Ctrl+N`, `Ctrl+S` and `Ctrl+W` shortcuts
- Do not require user to click "New" after starting the application

Fixes:

- Ensure the selected annotation point is removed. See #42
- Prevent downsizing of "New" icon
- Ensure interaction state is always up-to-date
- Improve state management of Apply, Reset and orientation sliders. Enable or disable the widget if appropriate.
- Ensure thickness is set after loading annotation
- Fix crash when loading -> closing scene -> loading

- Properly handle of view Reset without Apply
- Ensure stepSize slider singleStep is set
- Fix handling of interaction state to support "annotate" mode after loading

Cell Locator 0.1.0 2019-01-21

- Initial Release

1.7.7 Contributing to CellLocator

There are many ways to contribute to CellLocator.

- Submit a feature request or bug, or add to the discussion on the CellLocator issue tracker
- Submit a Pull Request to improve CellLocator.

The PR Process, and Related Gotchas

How to submit a PR ?

If you are new to CellLocator development and you don't have push access to the CellLocator repository, here are the steps:

1. Fork and clone the repository.
2. Create a branch.
3. Push the branch to your GitHub fork.
4. Create a Pull Request.

This corresponds to the `Fork & Pull Model` mentioned in the GitHub flow guides.

If you have push access to this repository, you could simply push your branch and create a Pull Request. This corresponds to the `Shared Repository Model` and will facilitate other developers to checkout your topic without having to configure a remote. It will also simplify the workflow when you are *co-developing* a branch.

When submitting a PR, make sure to add a `Cc: @cell-locator/developers` comment to notify CellLocator developers of your awesome contributions. Based on the comments posted by the reviewers, you may have to revisit your patches.

How to integrate a PR ?

Getting your contributions integrated is relatively straightforward, here is the checklist:

- All tests pass
- Consensus is reached. This usually means that at least one reviewer added a `LGTM` comment and a reasonable amount of time passed without anyone objecting. `LGTM` is an acronym for *Looks Good to Me*.

Next, there are two scenarios:

- You do NOT have push access: A CellLocator core developer will integrate your PR.
- You have push access: Simply click on the "Merge pull request" button.

Then, click on the "Delete branch" button that appears afterward.

1.7.8 Maintainers

Making a release

A core developer should use the following steps to create a release of **Cell Locator**

1. Update CHANGES.md adding missing entries in `Next Release` section
2. Remote connect to `overload` windows build machine (internally hosted at Kitware)
3. Open `Git Bash` and update source checkout

```
cd /d/D/P/CL-0
git pull origin main
```

4. Run `D:\D\DashboardScripts\overload-vs2019-cell-locator_preview_experimental.bat` script

   Scripts are available at https://github.com/Slicer/DashboardScripts

5. From the developer workstation, tag the release:

```
release=X.Y.Z-YYYY-MM-DD
git tag --sign -m "${release}" ${release} main
git push origin ${release}
```

   where:

   - `X.Y.Z` corresponds to the version in Applications/CellLocatorApp/slicer-application-properties.cmake
   - `YYYY-MM-DD` corresponds to the date of commit from which the application is built

   We recommend using a GPG signing key to sign the tag.

6. From the developer workstation:

- update CHANGES.md replacing `Next Release` with `Cell Locator X.Y.Z-YYYY-MM-DD` and push.
- update `AnnotationFileFormat.md` based on the Annotation File Format Versioning Guidelines and push.

7. Go to https://github.com/BICCN/cell-locator/tags, then create a *release* or *pre-release* from the tag.

   Update release description to include text like the following:

```
See [release notes](https://github.com/BICCN/cell-locator/blob/main/CHANGES.md#cell-
↪locator-XYZ-YYYY-MM-DD)
```

   where `XYZ-YYYY-MM-DD` should corresponds to the `release` set in previous steps.

8. From the build machine, login to GitHub and upload the package as a release asset.

1.8 Acknowledgments

1.8.1 Citation

If you use Cell Locator in your research, please cite the following Research Resource Identifier (RRID):

```
CellLocator (RRID:SCR_019264)
```

1.8.2 Funding Sources

This work was supported by the National Institute Of Mental Health of the National Institutes of Health under Award Numbers U01MH114812 (PI. E. Lein), U19MH114830 (PI. H. Zeng), and U24MH114827 (PIs M. Hawrylycz, L. Ng). The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

1.9 License

CHAPTER

TWO

INDICES AND TABLES

- genindex
- search

PYTHON MODULE INDEX

C

INDEX

Symbols

A

C

D

F

I

M

N

O

P

R

S

T

V