SlicerJupyter

A 3D Slicer kernel for interactive publications

Jean-Christophe Fillion-Robin @ Kitware

March 2nd, 2023

Software Development Panel: Web Visualization Frameworks

kitware

Scientific Computing and Imaging Institute, University of Utah

Speaker Background

- Distinguished Engineer @ Kitware in our North Carolina office
- Lead developer of 3D Slicer
- Maintainer of scikit-build, cmake, ninja and castxml python packages
- Maintainer of python-cmake-buildsystem
- Maintainer of dockcross

kitware

jcfr

Goals of this talk

- Explain why integrating traditional desktop application with Jupyter is relevant
- Describe how Jupyter ecosystem can leverage traditional desktop applications
- Present the 3D Slicer desktop application use case

Outline

- **Advantages of integrating Desktop Applications with Jupyter**
- **What is 3D Slicer ?**
  - capabilities, clinical use cases and community
  - building blocks
- **3D Slicer Embedded Python Interactor**
- **Slicer Jupyter Kernel:**
  - How it works?
  - Interactivity levels
- **Reproducing these slides / Demo**
- **What is next ?**

kitware

Problem statement (1 / 3)

Assuming your goal is to create a notebook to …

- manage a real-time time series of multimodal volumetric data,
- interactively define 3D regions of interest at each timepoint,
- and then calculate and visualize the nonlinear spatial mapping between them
- and display tables and charts of quantitative results.

How could you possibly do that??

Problem statement (2 / 2)

What if you …

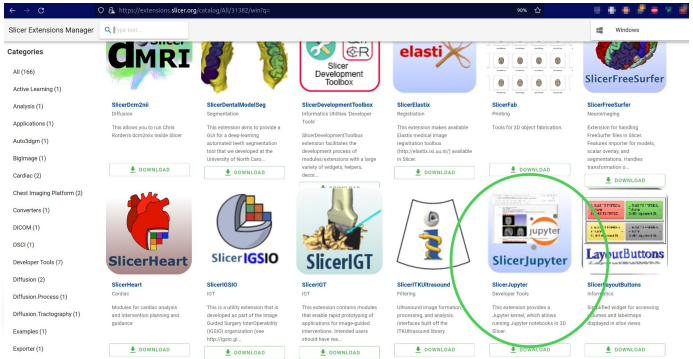
… found out there was already an actively maintained, open source, proven system that can do all of that and more?

… learned that it could be used in the Jupyter environment you already know and love?

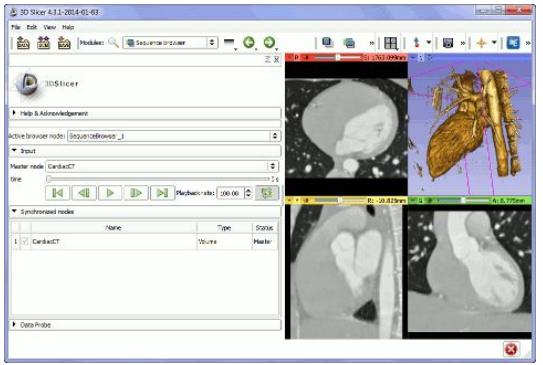
Proposed Solution

github.com/Slicer/SlicerJupyter#readme

- Guess what... it exists! and it's called SlicerJupyter!!!
- Do not redevelop everything from scratch but integrate.
- Leverage existing user and developer communities, share funding, etc.



Source: Wikipedia



Advantages

What the “desktop application” wins:

- Remains modern and relevant.
- The desktop application can run remotely, in the cloud, via web browser using a powerful development environment (not just a basic Python terminal)

What the “Jupyter ecosystem” wins:

- Tons of cool and relevant features, for free, immediately.

What is 3D Slicer ?

- Desktop application for medical, biomedical, and related imaging research
- Free and Open source [1]
- Cross-platform [2]

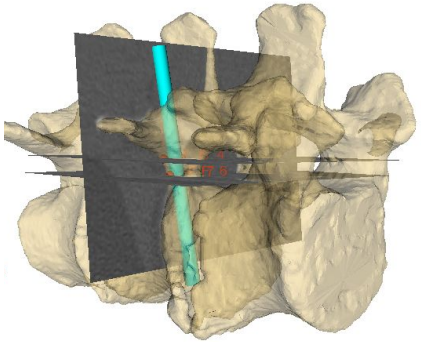
https://www.slicer.org

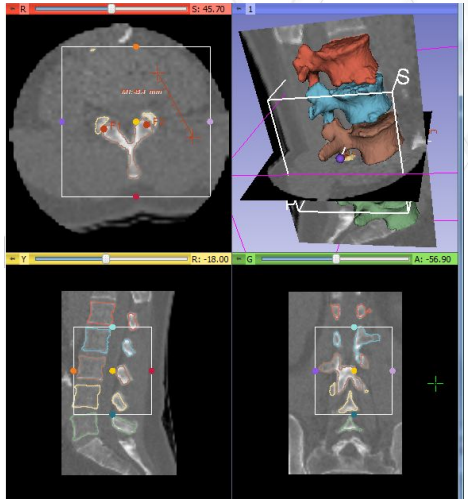
1. BSD-like license, completely free, non-restrictive license
2. Linux, macOS, Windows

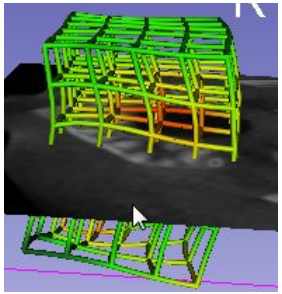
kitware

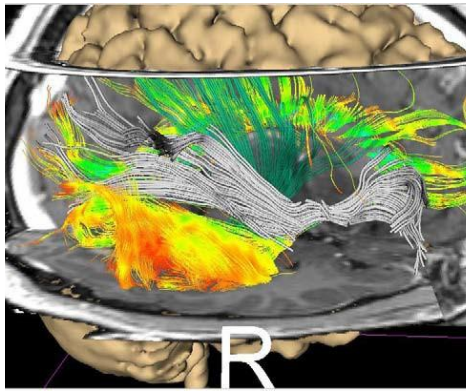
Capabilities

- DICOM standard interoperability
- Streamlined segmentation
- Artificial Intelligence
- 3D printing friendly
- 4D data support
- Python scripting
- Virtual Reality and Augmented Reality
- Surgical planning and guidance







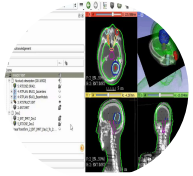


kitware

Slide: Credits Andras Lasso, The Perk Lab, Queens University

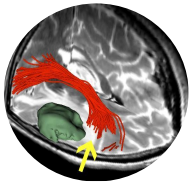
Clinical use cases

Radiation dose calculations



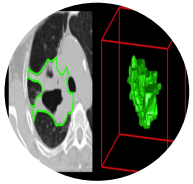
MRI-guided prostate biopsy

Tracking peritumoral white matter fibers

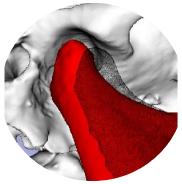


Breast cancer surgery guidance

Diagnosis of Different Tumors in Lung Cancer

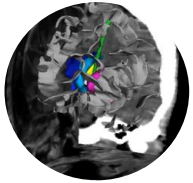


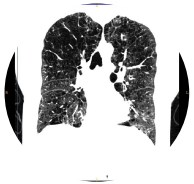
Clinical users drive creation of technology



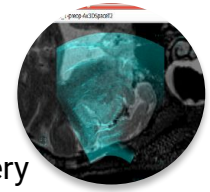
Diagnosis of Osteoarthritis Degeneration

Model-Guided Deep Brain Simulation

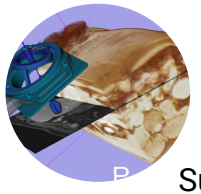




Quantitative assessment of COPD



Brain surgery

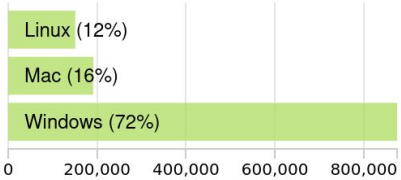


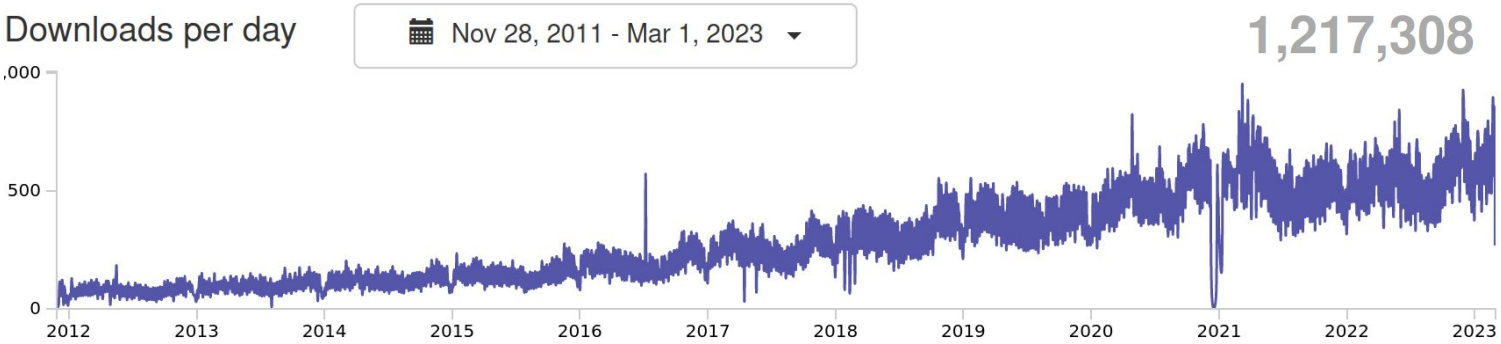
Surgical navigation

kitware

A large community

- **Since 1997 (first release)**
- **+$50M in funding (commercial & grants)**
- **Thousands of download per week, world wide**
  - 2012: 500 downloads per week
  - 2017: 2000 downloads per week
  - 2018: 3200 downloads per week
  - 2020: 3900 downloads per week
  - 2022: 4000 downloads per week





Sources:
https://download.slicer.org/download-stats/
https://slicer.readthedocs.io/en/latest/user_guide/about.html#funding-sources

kitware

Building Block: Hardware

software

hardware

kitware

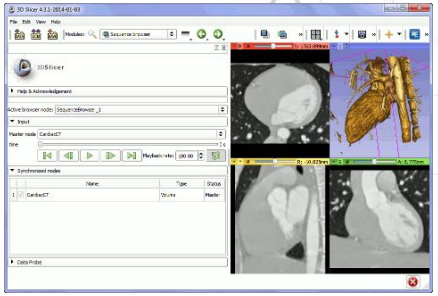
MRI, CT, PET scanners

Building Block: Application

Slide: Credits Andras Lasso, The Perk Lab, Queens University

Building Block: Extensions

… more than 150 extensions



Slicer extensions

3D Slicer

Visualization | Registration | Segmentation | … | Quantification

VTK, ITK, CTK, QT, Python, DCMTK, …

DICOM

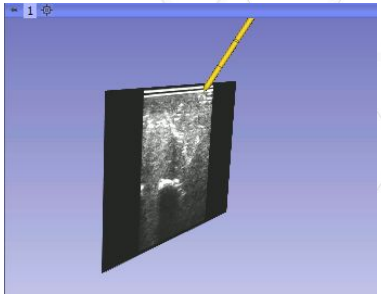
software

hardware

kitware

MRI, CT, PET scanners

Example: Real time surgical guidance



Slicer extensions

3D Slicer SDK

Visualization | Registration | Segmentation | ... | Quantification

VTK, ITK, CTK, QT, Python, DCMTK, …

OpenIGTLink | DICOM

PLUS

Calibration, synchronization, pre-processing, simulation, record&replay

OpenIGTLink

Device interface

software

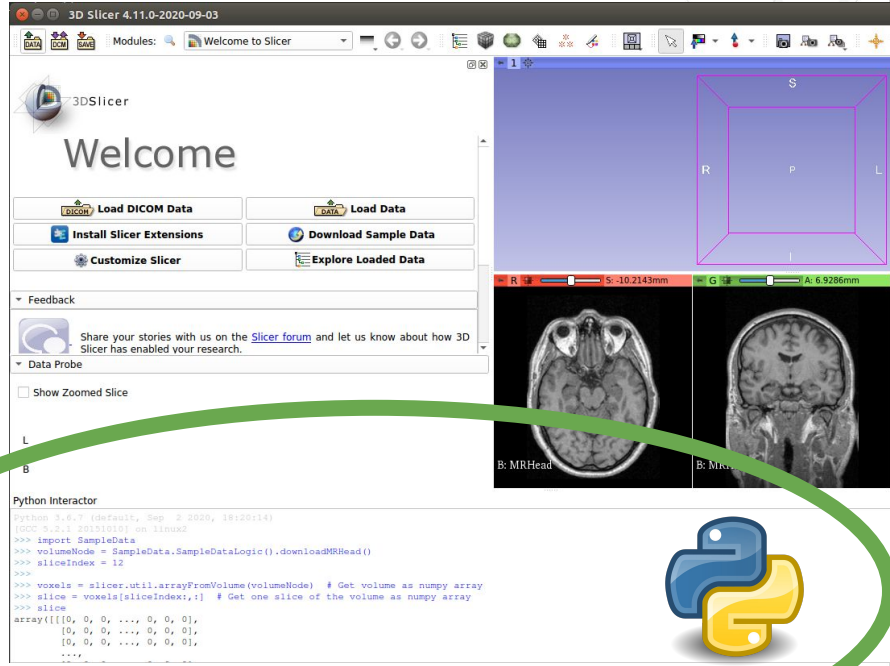
hardware

kitware

Surgical microscopes, endoscopes | Ultrasound scanners | Navigation systems | Robotic devices, manipulators | MRI, CT, PET scanners

Motivation for an “Embedded Python Interactor”

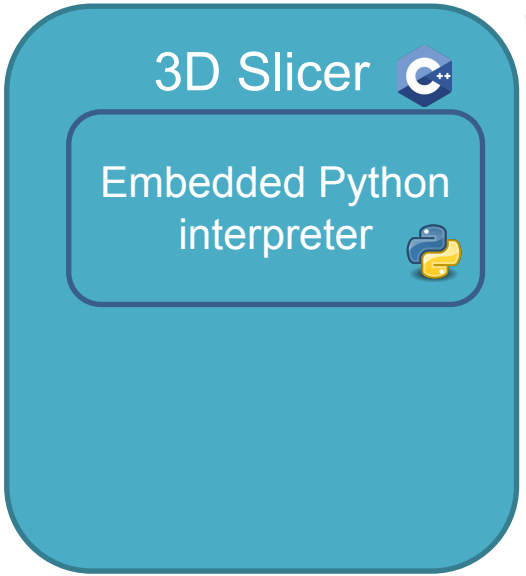
- Integration of all the building blocks is non-trivial
- Users can not pick and choose all the component
- 3D Slicer provide a rich Python environment



kitware

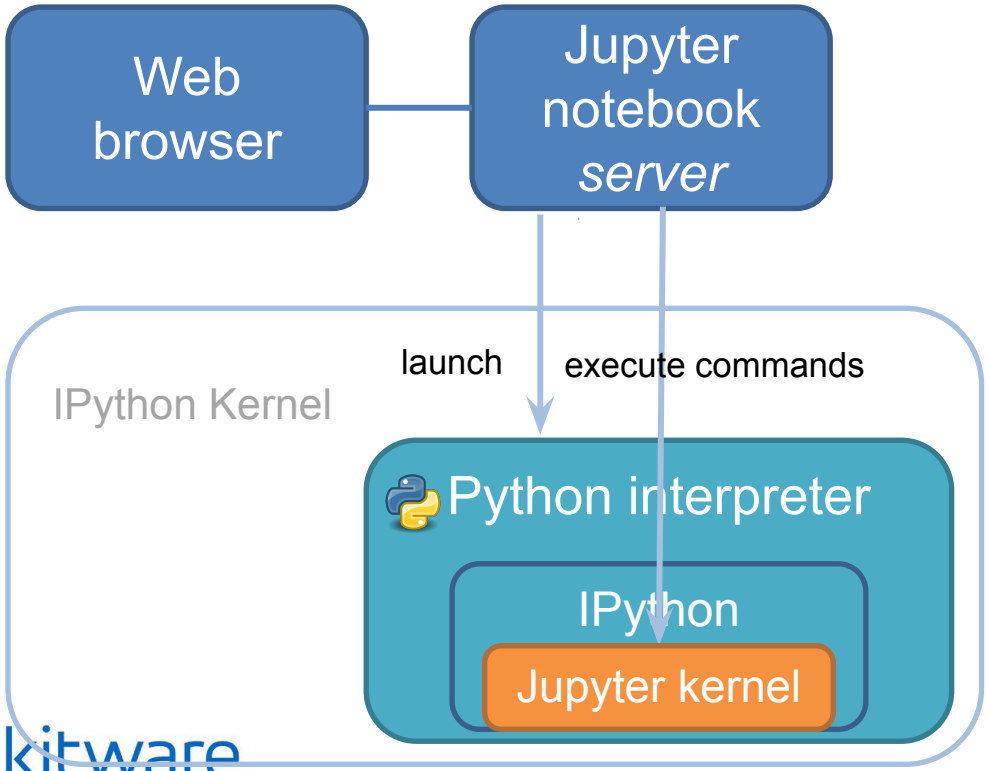
3D Slicer Embedded Python Interactor

- **Python 3**
- **Pip install of any Python packages is essential.**
- **Qt integration with PythonQt**
  - **Full control over our event loop**
- **3D Slicer extension**
  - bundle python packages
  - Or download at runtime



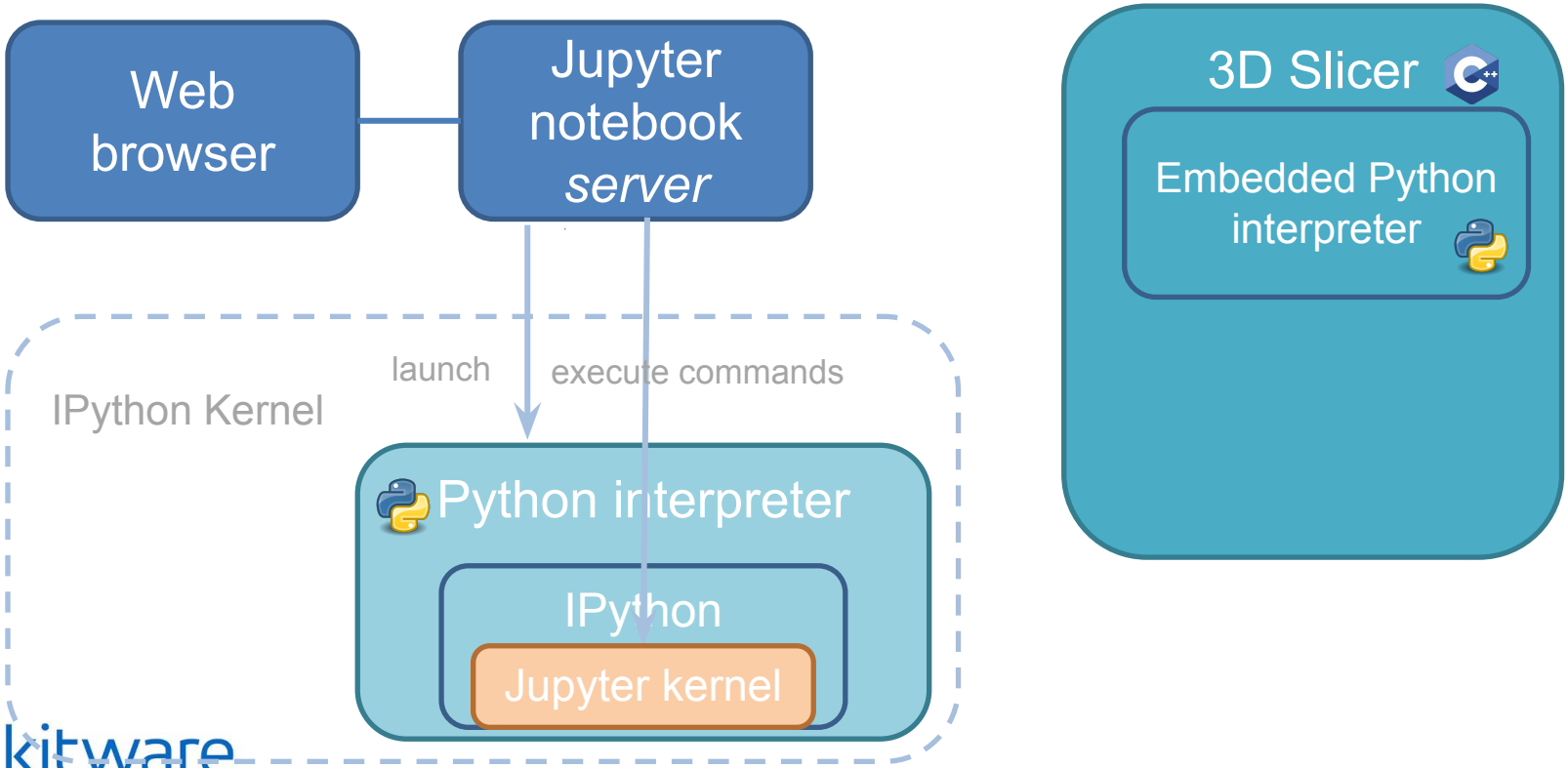
kitware

IPython Kernel



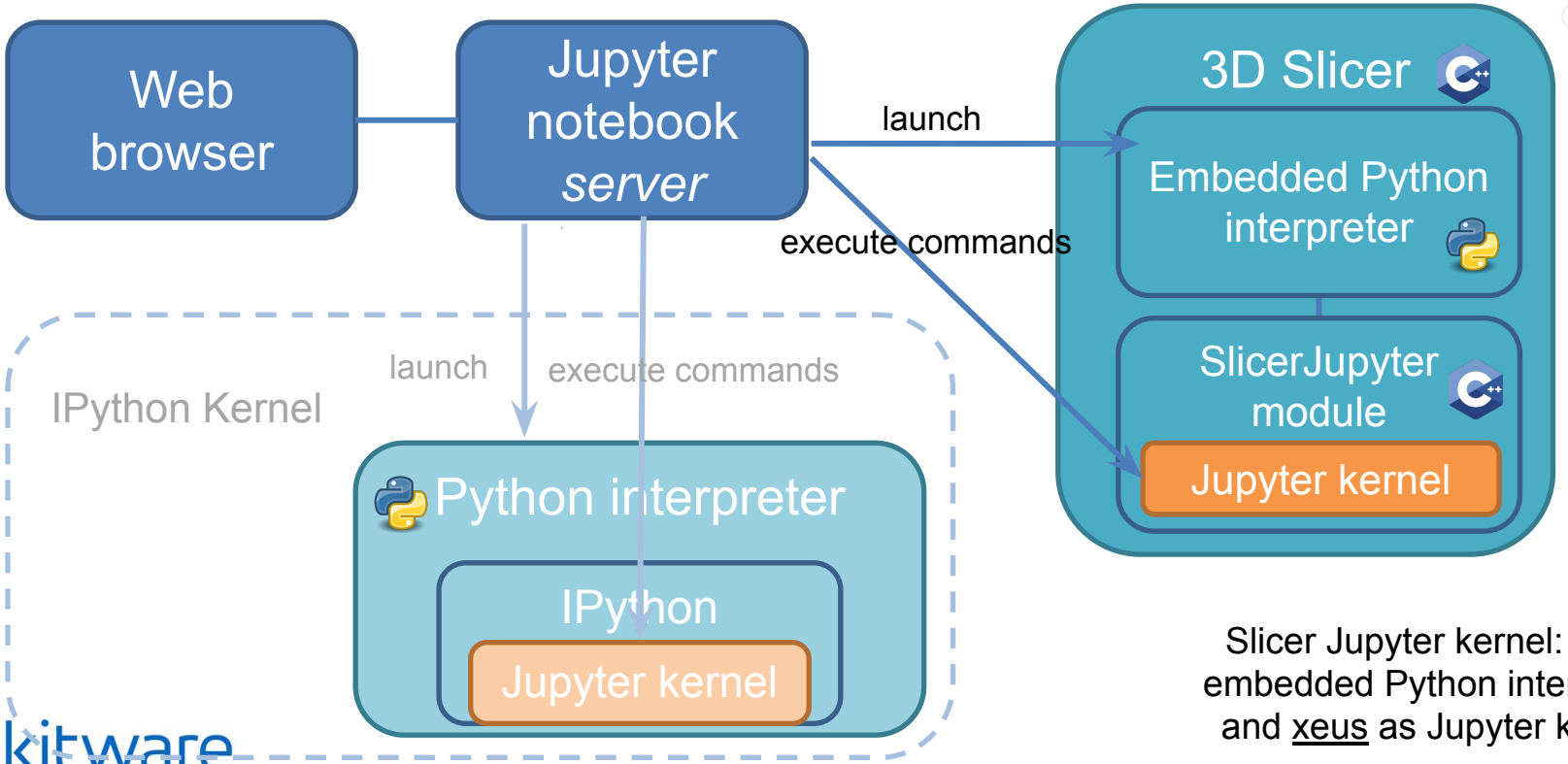
kitware

Slicer Jupyter Kernel: How it works? (1 / 2)



kitware

Slicer Jupyter Kernel: How it works? (2 / 2)



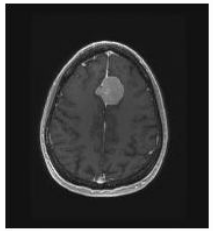
Slicer Jupyter kernel: uses embedded Python interpreter and xeus as Jupyter kernel

Interactivity: View objects + standard widgets (Level 1)

- Displayed content is saved in the notebook
- Views cannot be placed in a layout
- Low update rate (only for small adjustment of view parameters)
- Mouse and keyboard events are not captured

In [7]:
```
# Slice view display
from ipywidgets import interact
@interact(position=(0,100))
def update(position=50):
    return slicernb.ViewSliceDisplay('Red', positionPercent=position)
```

position 69



[9]:
```
# Enable volume cropping
displayNode = slicer.modules.volumerendering.logic().GetFirstVolumeRenderingDisplayNode(volume)
displayNode.SetCroppingEnabled(True)
slicer.modules.volumerendering.logic().CreateROINode(displayNode)
roiNode = displayNode.GetROINode()

# 3D view display
slicernb.reset3DView()
roiNode.GetDisplayNode().SetVisibility(False)  # hide the ROI box
from ipywidgets import interact
@interact(roll=(-90.0,90.0,5), pitch=(-90.0,90.0,5), yaw=(-180.0,180.0,5), cropx=(0,140,5), cropy=(0,240,5), cropz=(0,160,5))
def update(roll=0, pitch=-10, yaw=30, cropx=140, cropy=240, cropz=160):
    #roiNode.SetRadiusXYZ([cropx, cropy, cropz])
    roiNode.SetSizeWorld([cropx, cropy, cropz])
    return slicernb.View3DDisplay(0, orientation=[roll, pitch, yaw])
```

roll 0.00

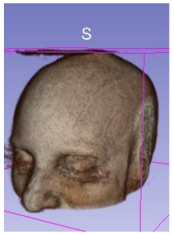
pitch −10.00

yaw 30.00

cropx 140

cropy 240

cropz 160

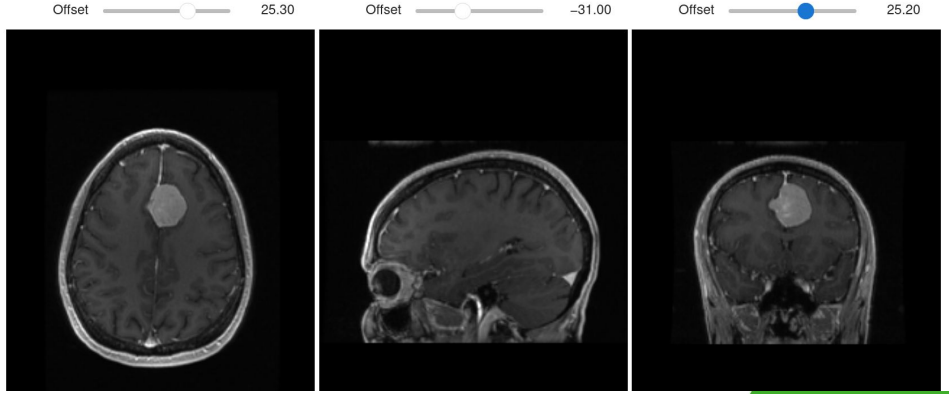


kitware

Interactivity: View widgets (Level 2)

- Widgets can be placed in a layout
- Widget state (displayed content) is not saved in the notebook by default
- Low update rate (only for small adjustment of view parameters)
- Mouse and keyboard events are not captured

```
[14]: # Slice widgets
import JupyterNotebooksLib as slicernb
from ipywidgets import HBox
display(HBox([slicernb.ViewSliceWidget('Red'), slicernb.ViewSliceWidget('Yellow'), slicernb.ViewSliceWidget('Green')]))
```



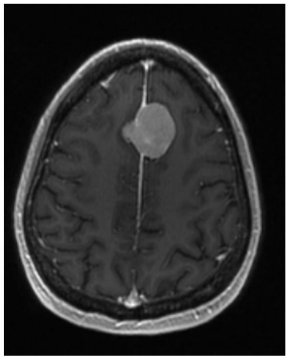
kitware

Interactivity: Interactive view widgets (Level 3)

- Some view controlling mouse and keyboard events are captured
- Only selected view can be displayed and controlled
- Medium update rate (somewhat usable on remote computers)

[16]:
```
# Adjust maximum rate of Slicer's Jupyter kernel consuming Jupyter messages.
# Lower values make the notebook more responsive but too low values may make the Slicer application
# slow to respond.
slicer.modules.jupyterkernel.setPollIntervalSec(0.001)

# 3D view
slicernb.AppWindow.setWindowSize(scale=0.8)
live3d = slicernb.ViewInteractiveWidget('1')
live3d.trackMouseMove = True
display(live3d)
```

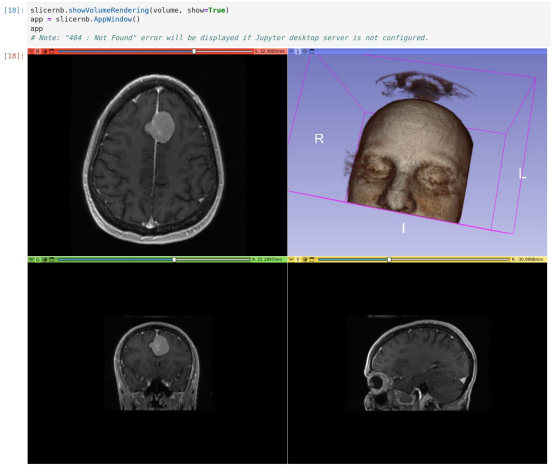
[17]:
```
# Slice view (use arrow keys to move between slices, right-click-and-drag to zoom in/out)
liveRedSlice = slicernb.ViewInteractiveWidget('R')
liveRedSlice.trackMouseMove = True
display(liveRedSlice)
```

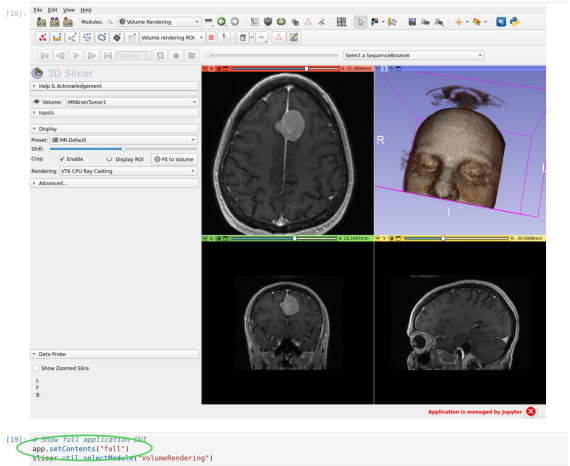


kitware

Interactivity: Remote application window view (Level 4)

- High update rate (suitable for working on remote computers)
- All mouse and keyboard events are captured
- Full application window can be displayed and controlled
- Only available if Jupyter desktop server is configured





Rich Display Support

- Markups, Model, Transforms, …
- See [JupyterNotebooksLib/display.py]

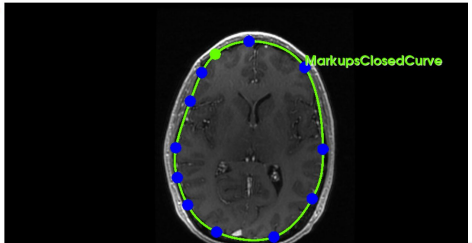
```
[15]:
import numpy as np
markupPoints = np.array([
    [ 69.12484176,  -8.73226641,  -2.175     ],
    [ 56.3258959 ,  32.61817407,  -2.175     ],
    [ 46.15237483,  57.88788769,  -2.175     ],
    [ 34.02739799,  74.35843751,  -2.175     ],
    [  3.93193405,  85.64423649,  -2.175     ],
    [-45.51061385,  62.53521954,  -2.175     ],
    [-61.63318381, -10.01634531,  -2.175     ],
    [-51.97287678, -54.02084408,  -2.175     ],
    [-18.1022449 , -87.94210015,  -2.175     ],
    [ 32.95255999, -83.64274816,  -2.175     ],
    [ 58.6231426 , -59.59987177,  -2.175     ],
    [ 67.88479492, -35.27503826,  -2.175     ]])

closedCurve = slicer.mrmlScene.AddNewNodeByClass("vtkMRMLMarkupsClosedCurveNode")
slicer.util.updateMarkupsControlPointsFromArray(closedCurve, markupPoints)

slicernb.displayable(closedCurve)
```

[15]:

	label	position.R	position.A	position.S	selected	visible	description
0	MarkupsClosedCurve-1	69.124842	-8.732266	-2.175	True		
1	MarkupsClosedCurve-2	56.325896	32.618174	-2.175	True		
2	MarkupsClosedCurve-3	46.152375	57.887888	-2.175	True		
3	MarkupsClosedCurve-4	34.027398	74.358438	-2.175	True		
4	MarkupsClosedCurve-5	3.931934	85.644236	-2.175	True		

```
[16]:
closedCurve.SetNthControlPointSelected(3, False)
closedCurve.GetDisplayNode().SetSelectedColor(0,0,1)
closedCurve.GetDisplayNode().SetGlyphScale(5)
closedCurve.GetDisplayNode().UseGlyphScaleOn()
slicer.util.setSliceViewerLayers(fit=True)
slicer.modules.markups.logic().JumpSlicesToNthPointInMarkup(closedCurve.GetID(), 1)
slicernb.AppWindow.setWindowSize(scale=0.5)
slicernb.AppWindow.setContents("viewers")
slicernb.ViewDisplay('OneUpRedSlice', center=False)
```

[16]:

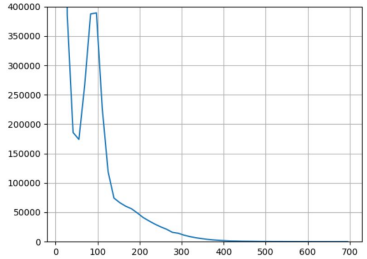



```
[17]:
# Simple static display

modelNode=slicer.modules.models.logic().AddModel(slicernb.localPath("data/ProstateMeanShape.stl"))
slicernb.displayable(modelNode)
```

[17]:

```
[19]:
try:
  import matplotlib
except ModuleNotFoundError:
  slicer.util.pip_install('matplotlib')
  import matplotlib

matplotlib.use('Agg')

# Get a volume from SampleData and compute its histogram
import numpy as np
histogram = np.histogram(slicer.util.arrayFromVolume(volume), bins=50)

# Show a plot using matplotlib
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.plot(histogram[1][1:], histogram[0].astype(float))
ax.grid(True)
ax.set_ylim((0, 4e5))

slicernb.MatplotlibDisplay(plt)
```

[19]:



kitware

Improved Tutorial Experience

Problem solved with Jupyter integration:

- Always switch context between web browser/PDF viewer and application
- Difficult for users to share their own workflow
- Training material hard to maintain and test

Reproducing these slides / Demo

- Interactivity levels
- Segmentation workflow

https://mybinder.org/v2/gh/Slicer/SlicerNotebooks/master

kitware

Acknowledgments & Thank you

- Scientific Computing and Imaging Institute
  University of Utah
- National Institute of General Medical Sciences (NIGMS) of the National Institutes of Health (NIH)
  R24 GM136986
- See also https://slicer.readthedocs.io/en/latest/user_guide/about.html#funding-sources

Alphabetically ordered by lastname


Sylvain Corlay


J-Christophe Fillion-Robin


Mike Grauer


Andras Lasso


Johan Mabille


Matt McCormick


Isaiah Norton


Steve Pieper


Martin Renou


Mike Sarahan

Questions ?

Getting Started:
github.com/Slicer/SlicerJupyter#readme

Learn more:
blog.jupyter.org/slicerjupyter-a-3d-slicer-kernel-for-interactive-publications-6f2ad829f635

kitware

What is next ?

- Create Slicer tutorials as Jupyter notebooks
- JupyterLab debugger support
- Improved JupyterLab panel usage (Slicer views displayed as JupyterLab panels)
- Create demos using Voilà

Everything Python: Avoiding the temptation ....

There may be a temptation

to develop everything from scratch in Python with Jupyter

but

this would be lots of wasted effort

and create competition between developers

instead of collaboration.

Source: Wikipedia

kitware