



CSIM – Computação em
Sinais e Imagens Médicas

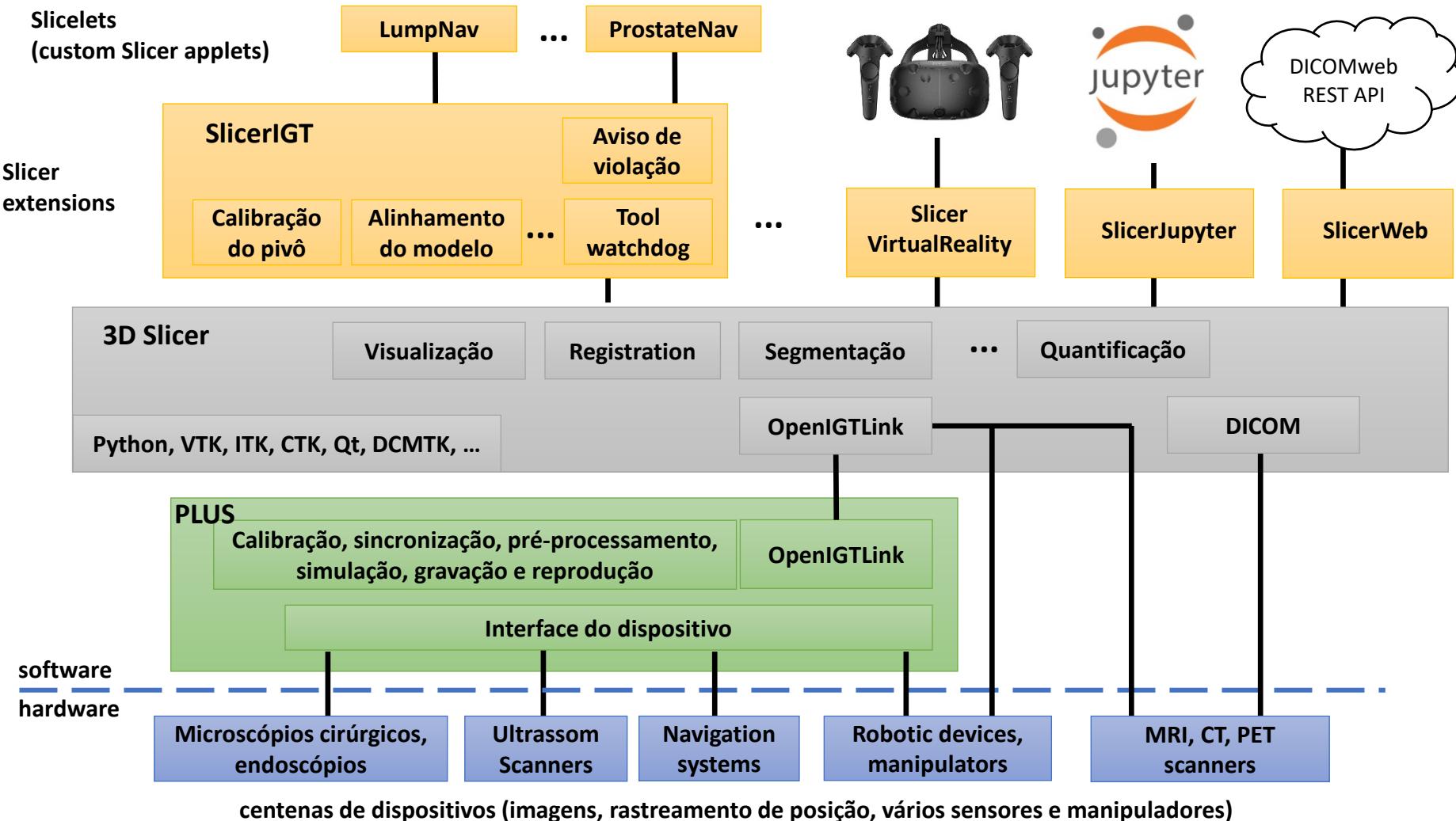
3D Slicer

Visão geral

&

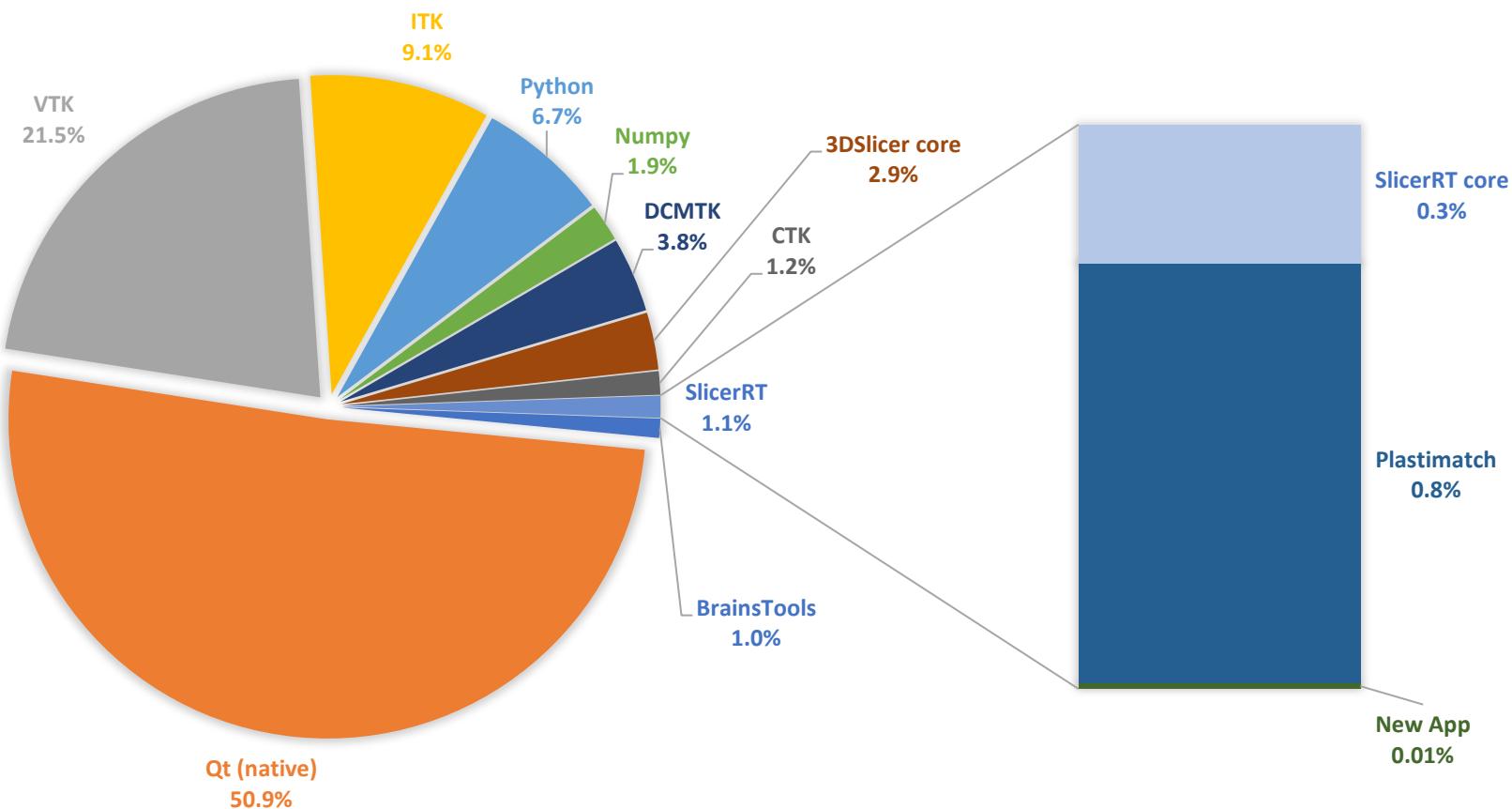
Noções básicas de programação

3D Slicer visão geral da plataforma



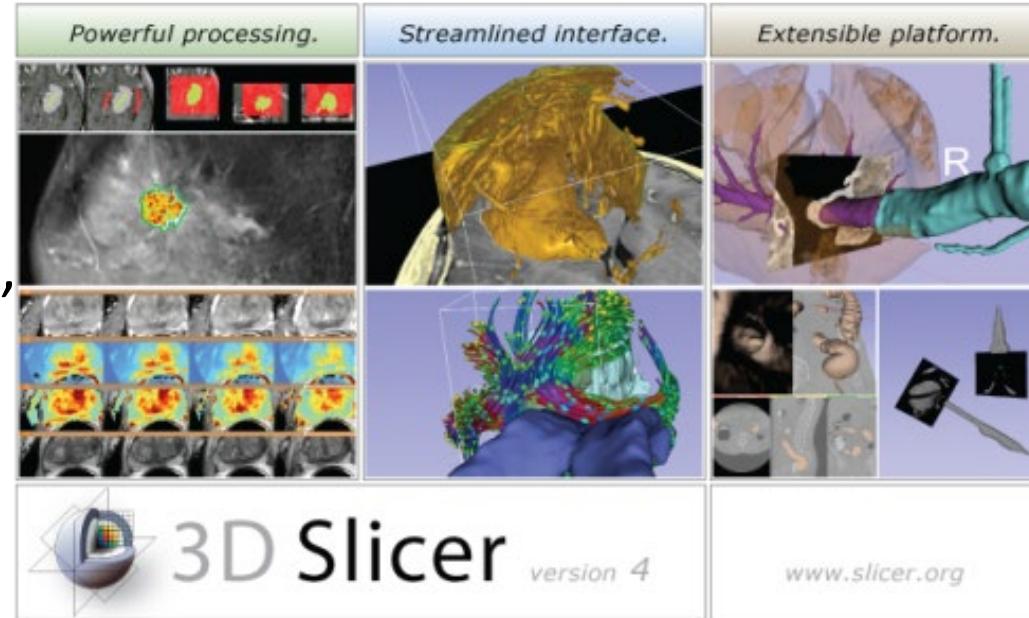
Construindo a plataforma

LINHAS DE CÓDIGO-FONTE EM UM APLICATIVO BASEADO NO 3D SLICER



Características do 3D Slicer

- Aplicação de software para computação de imagens médicas: importação/exportação de dados, visualização, segmentação, registro, quantificação, orientação em tempo real
- Estrutura do aplicativo: módulos personalizados personalizáveis e extensíveis
- Completamente gratuito (BSD)
- Multi-plataforma



- Suporte ao usuário e ao desenvolvedor
- Cursos de formação, documentação, tutoriais

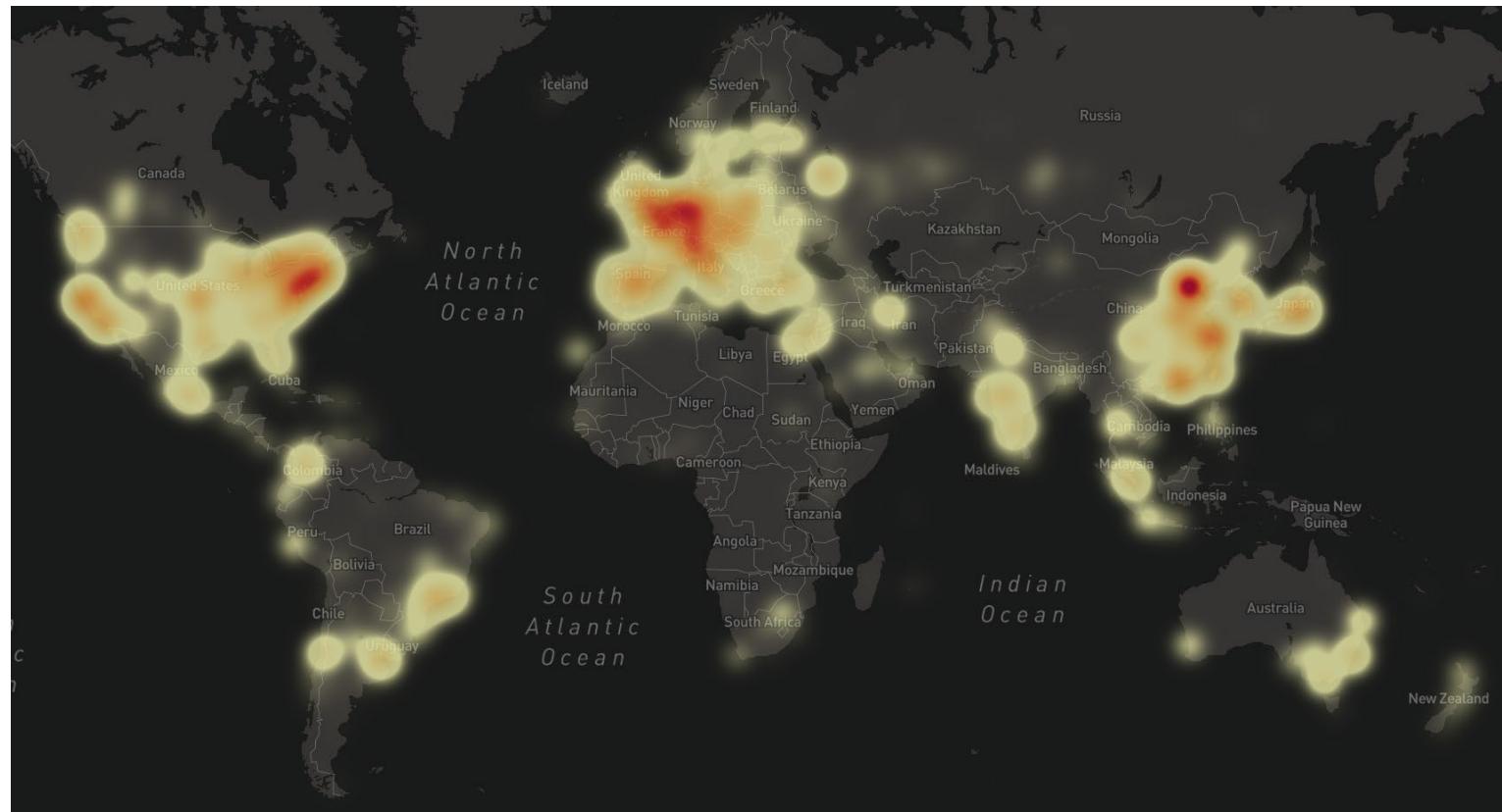
Fedorov, et al. "3D Slicer as an image computing platform for the Quantitative Imaging Network." Magnetic resonance imaging 30.9 (2012): 1323-1341.

Grande comunidade de usuários

500 downloads por semana em 2012

3700 downloads por semana em 2022

770 000+ downloads nos últimos 5 anos:



Semana de projeto “Project week”

- Duas vezes por ano
- Traga seu próprio projeto, trabalhando com especialistas
- Reuniões, formação
- Próximo:
Verão 2023, evento híbrido
<https://projectweek.na-mic.org/>



3D Slicer em uso clínico



Uso comercial (licensa BSD)



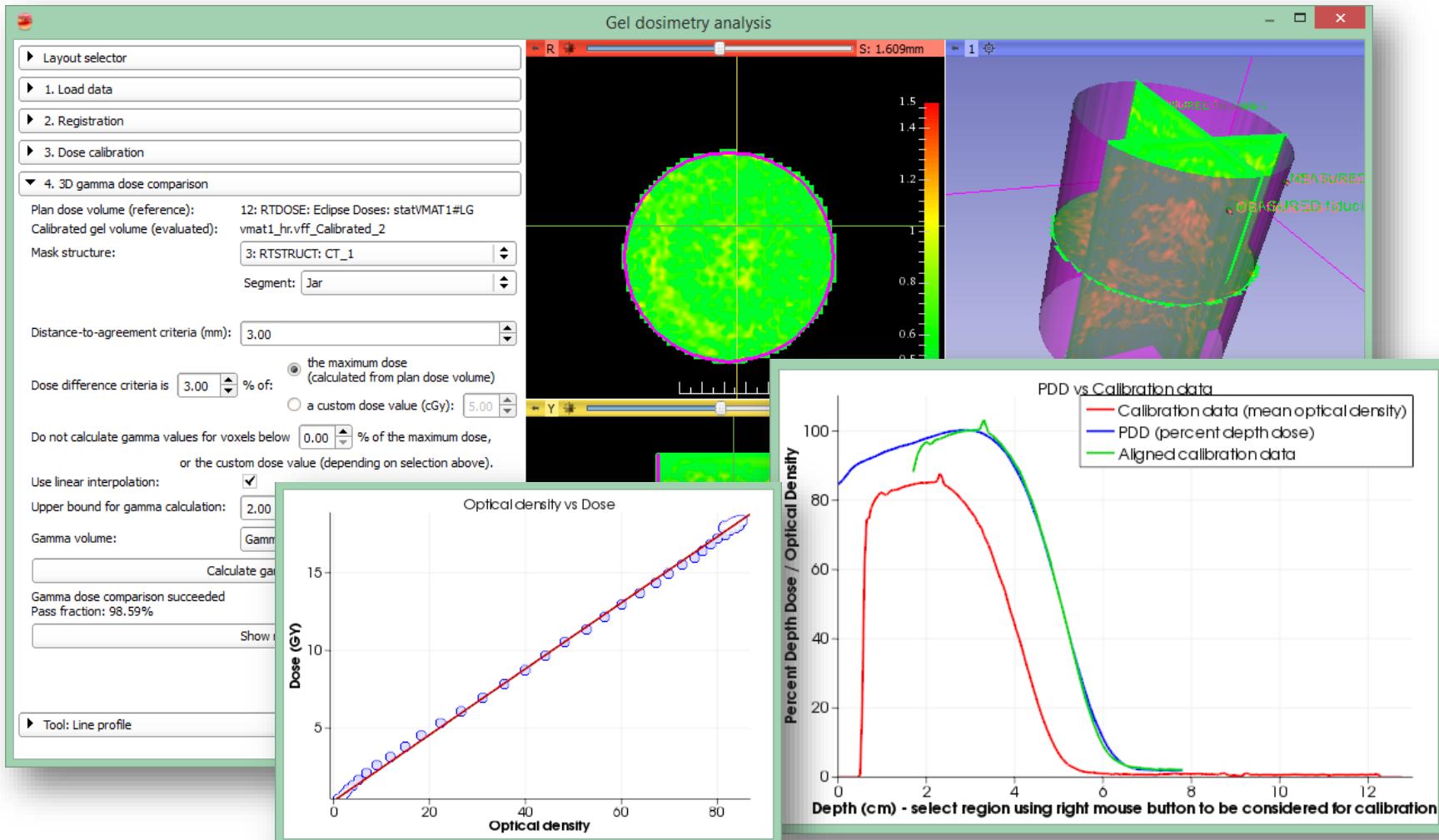
Known commercial activities range from use “as is” to full blown product development:



- Xstrahl (small animal radiation product)
- mebio (radiology product, prostate guidance)
- SonoVol (ultrasound product) (R43CA192482...)
- Novartis (quantitative imaging clinical trials)
- New Frontier (navigation system)
- KUKA (surgical robotics)
- Siemens (diagnostic and interventional research)
- Canon (robotic interventions)
- GE (research and products)
- NDI (trackers for surgical navigation)
- Isomics (research, consulting)
- Kitware (research, consulting)
 - 10+ Slicer based projects in the past two years
 - 5 commercial products being launched

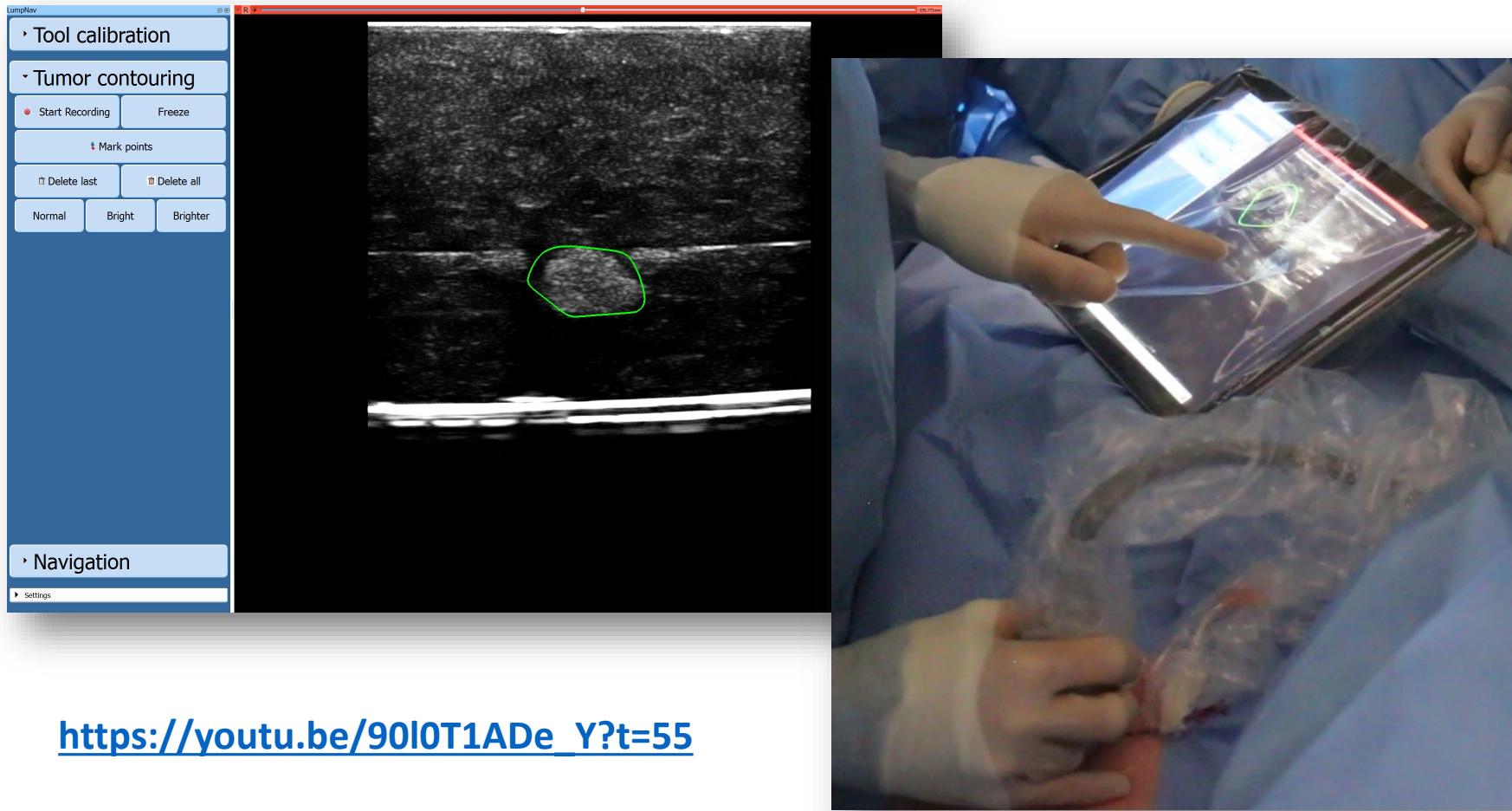


Exemplo: ferramenta Dosimetria em Gel



<https://www.slicer.org/slicerWiki/index.php/Documentation/Nightly/Modules/GelDosimetry>

Exemplo: LumpNav (otimizado para toque)



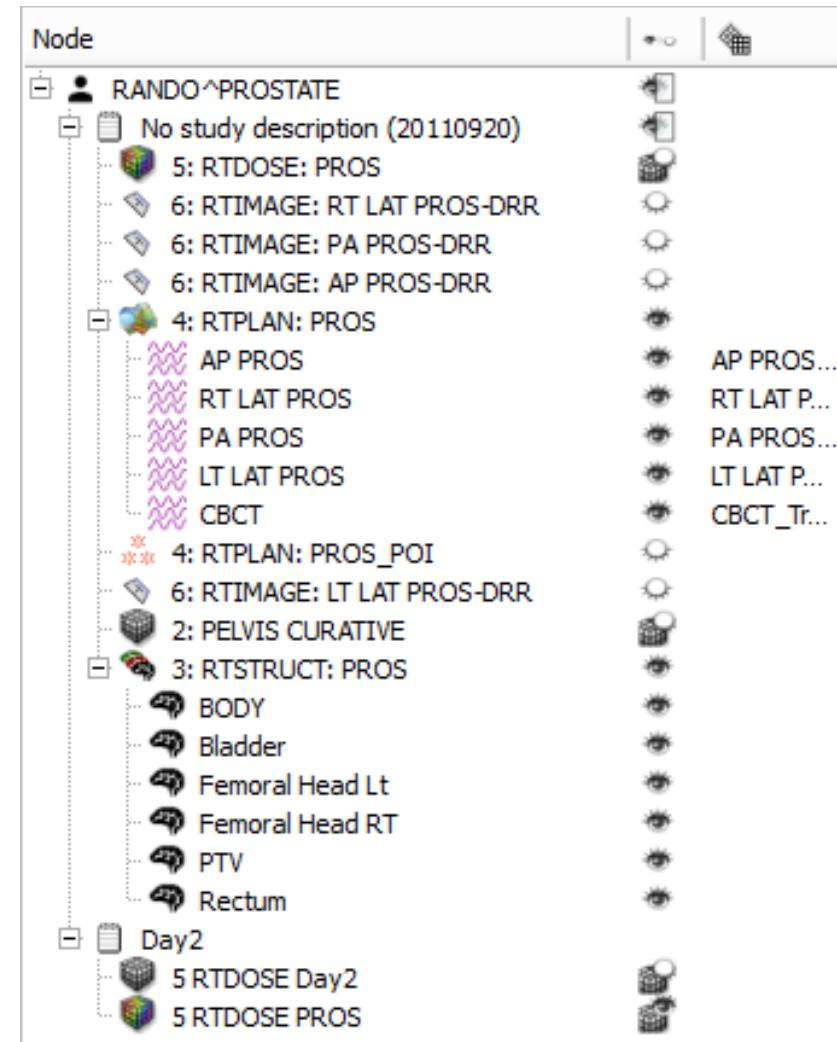
https://youtu.be/90I0T1ADe_Y?t=55

<http://www.slicerigt.org/wp/breast-cancer-surgery/>

Opção escolhida: "janela principal simplificada"

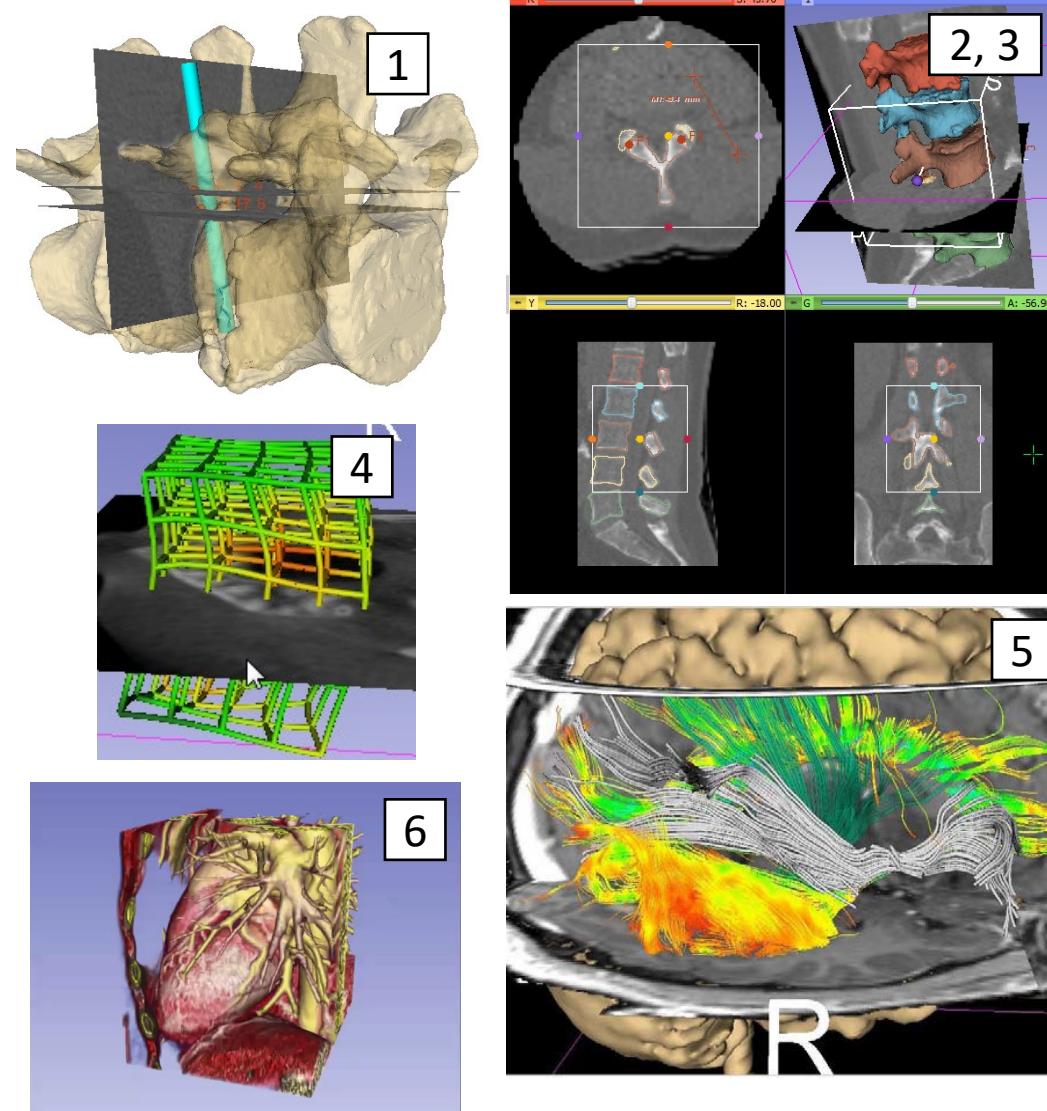
Importação/exportação de dados

- DICOM: volumes 2D/3D/4D, conjuntos de estruturas, volumes de dose, etc. (extensível sem Alterações do núcleo da segmentação de dados)
- Formatos de dados de pesquisa para volumes, malhas, transformações (NRRD, MetalO, VTK, HDF, etc.)
- Formatos de dados não médicos comuns (JPEG, TIFF, etc.)
- Salva e conclui a restauração do estado do aplicativo



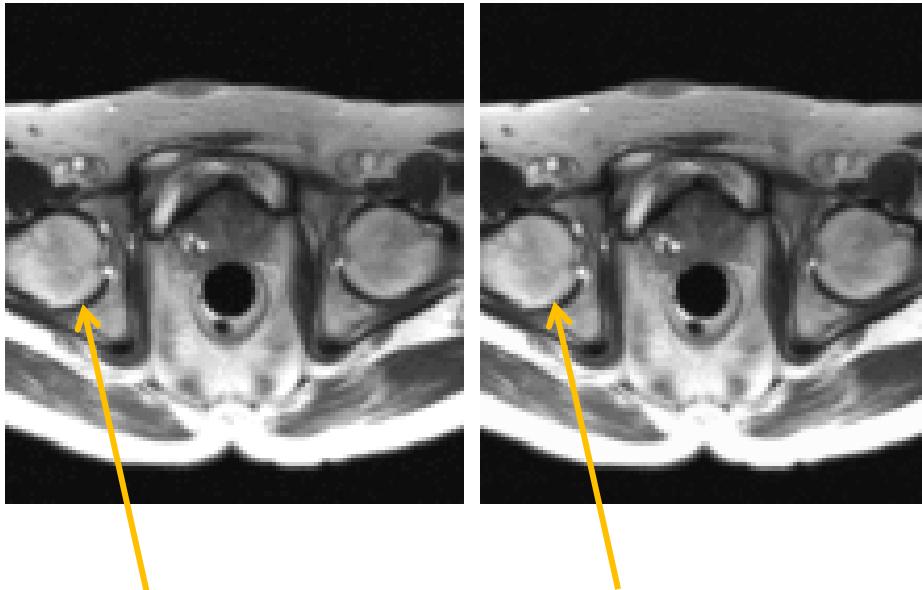
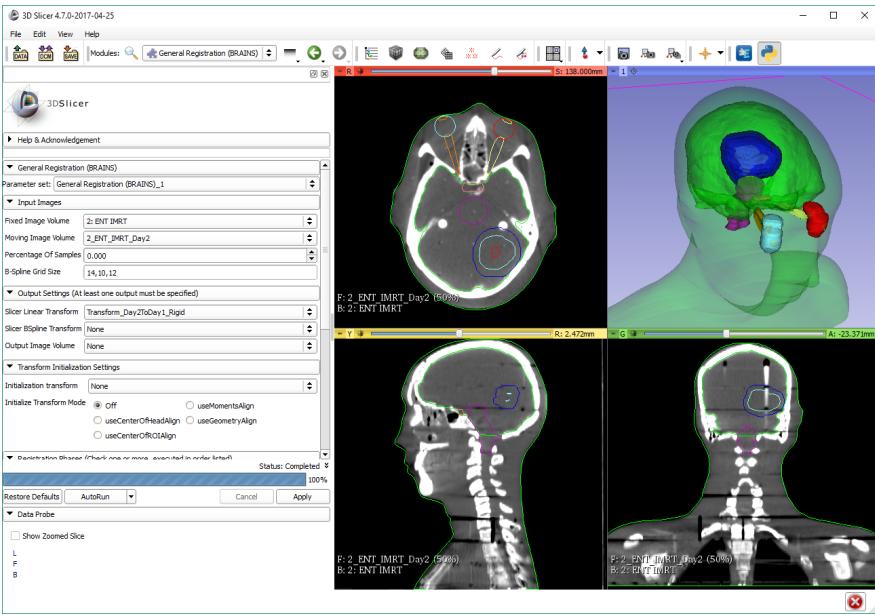
Visualização

1. 2D (fatia) e visões 3D, painéis de gráficos
2. Layout configurável
3. Fusão de imagens multimodalidade (primeiro plano, plano de fundo, mapa de rótulos)
4. Transformações, visualização de campos vetoriais e tensoriais
5. Renderização de superfície e volume
6. Dados de sequência temporal



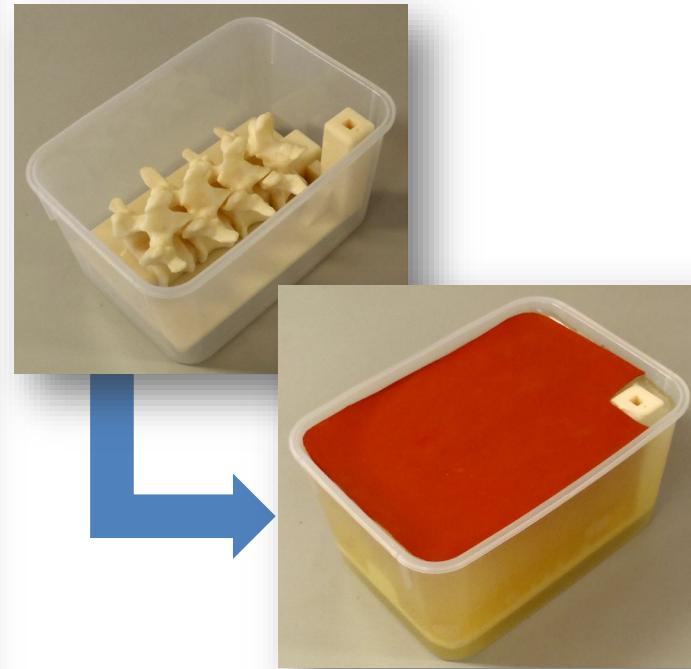
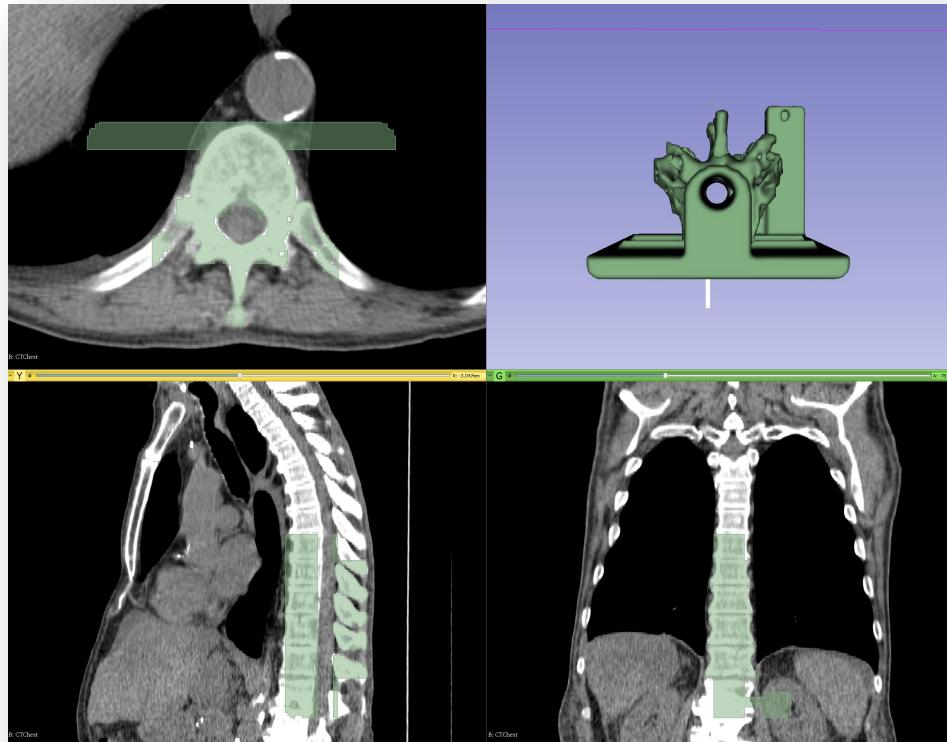
Alinhamentos

- Manual: translação, rotação em 3D
- Automático: rígido, deformável, com várias métricas de similaridade, métodos de inicialização, otimizadores, mascaramento, etc.
- Extensões: registro baseado em estrutura, Elastix, etc.



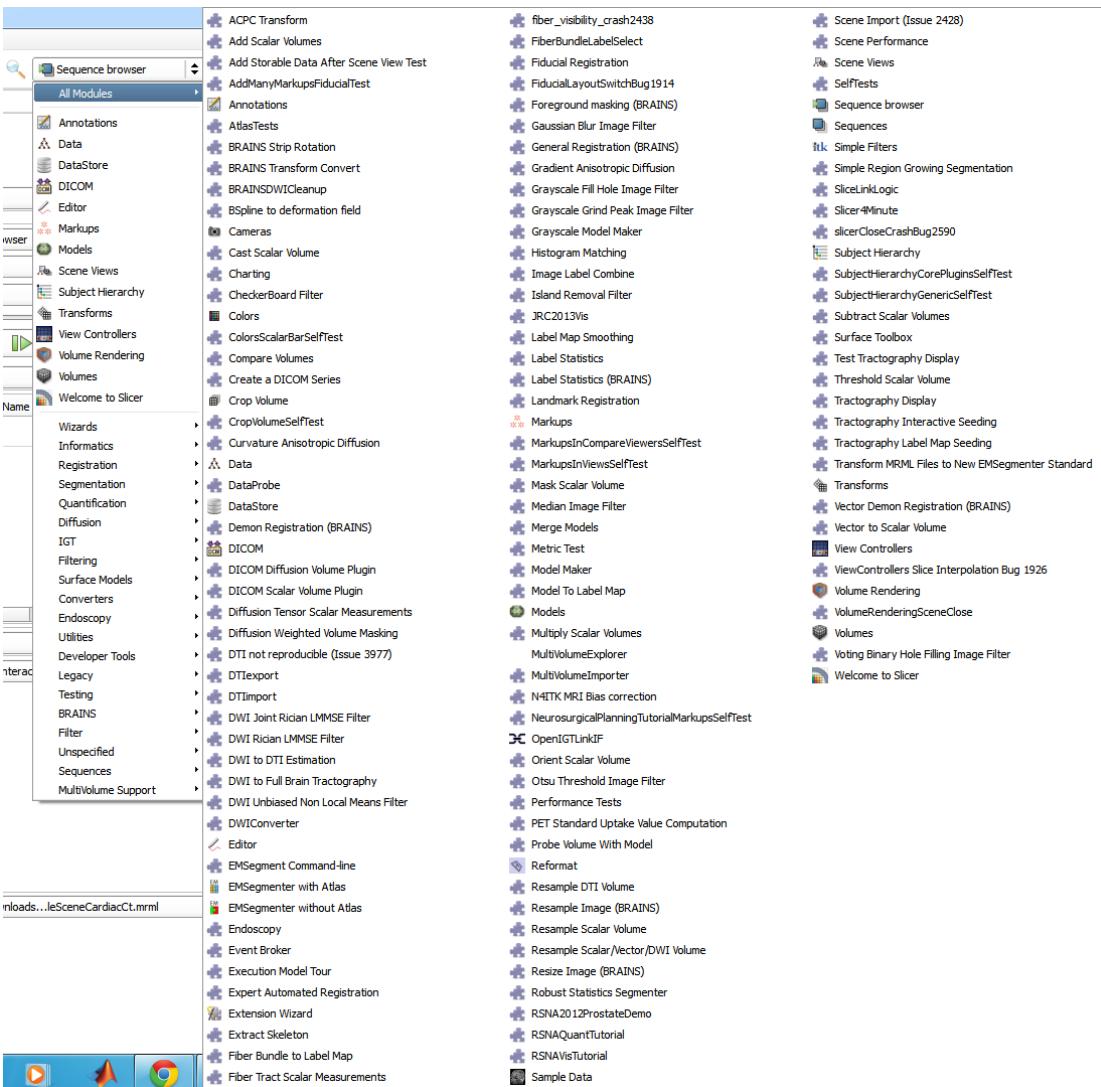
Segmentações

- Manual (pintura, desenho, tesoura, limiar, etc.)
- Semiautomático (crescimento de região, preenchimento entre fatias, etc..)
- Automático (baseado em atlas, estatísticas robustas, etc..)



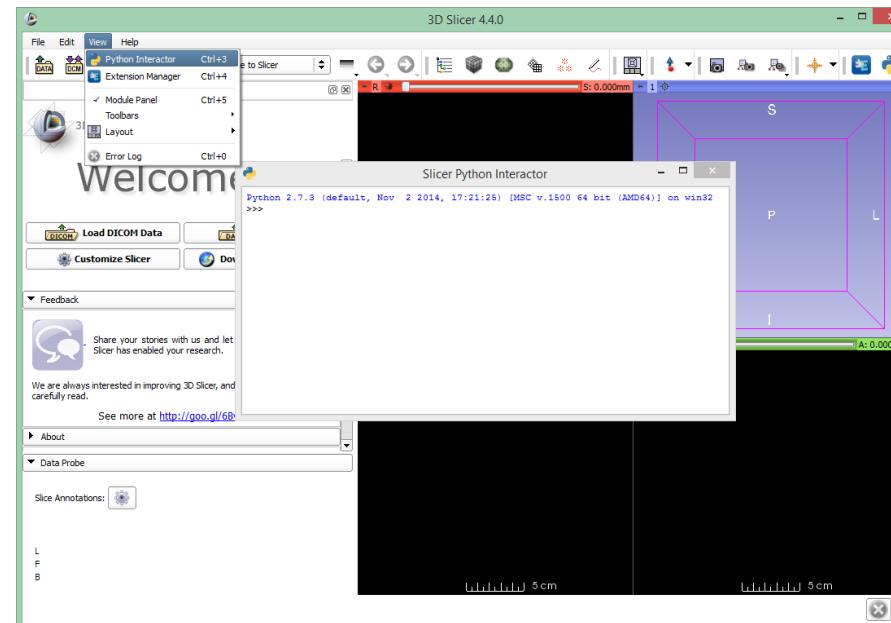
Muitos outros módulos...

- Filtragem de imagens (redução de ruído da imagem, correção do viés de ressonância magnética, etc.)
- Processamento de superfície
- Imagem de difusão
- Quantificação, estatística
- ...

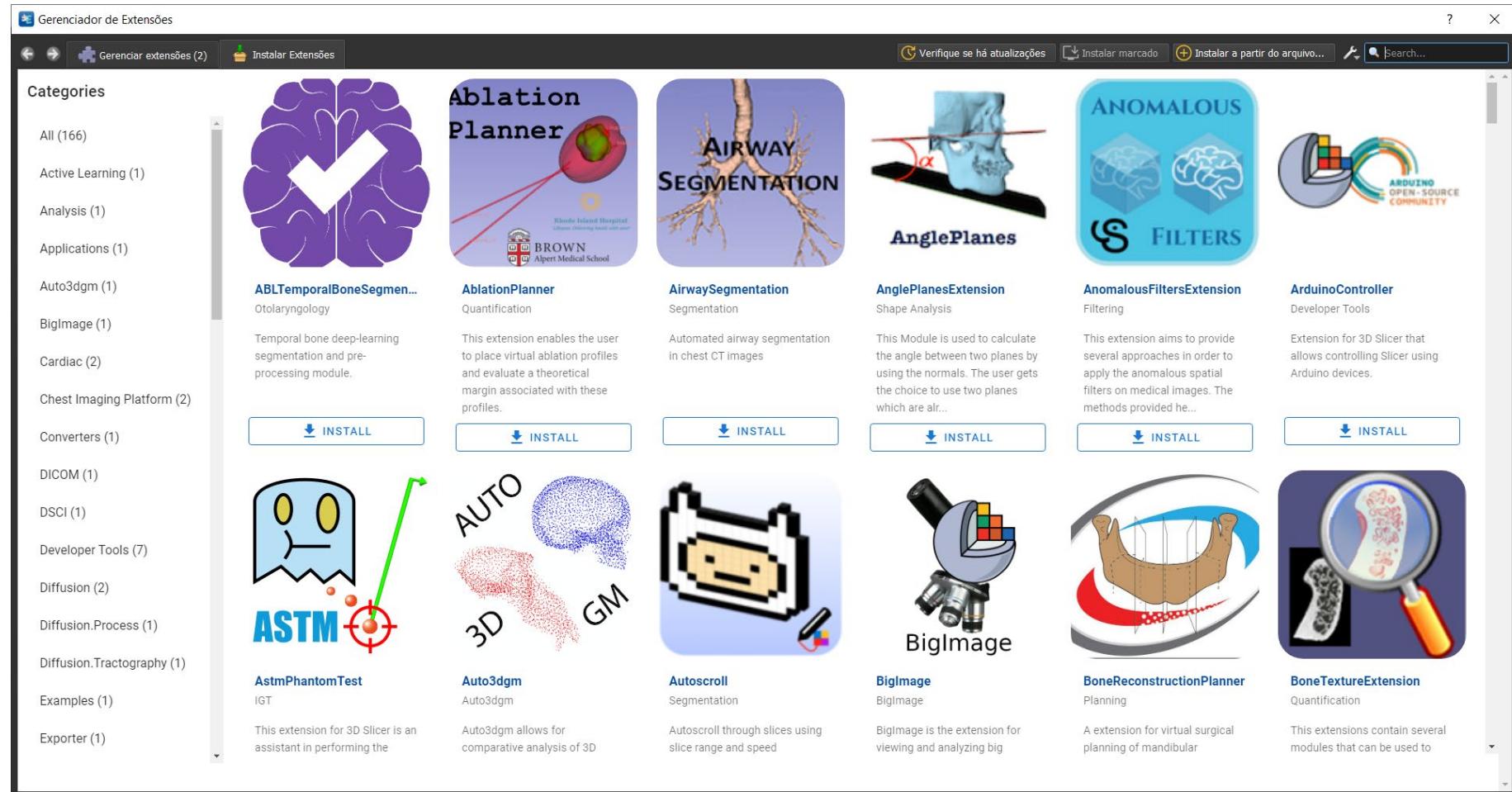


Python no Slicer

- Todos os dados, processamento, elementos da interface do usuário são acessíveis a partir do Python
- O console interativo do Slicer dá acesso a todos os objetos de dados e elementos da interface do usuário
- Qualquer pacote Python pode ser instalado via pip e usado



Slicer é extensível



O Gerenciador de Extenções do Slicer (“Extension Manager”) oferece ao usuário a possibilidade de baixar e instalar módulos adicionais do Slicer

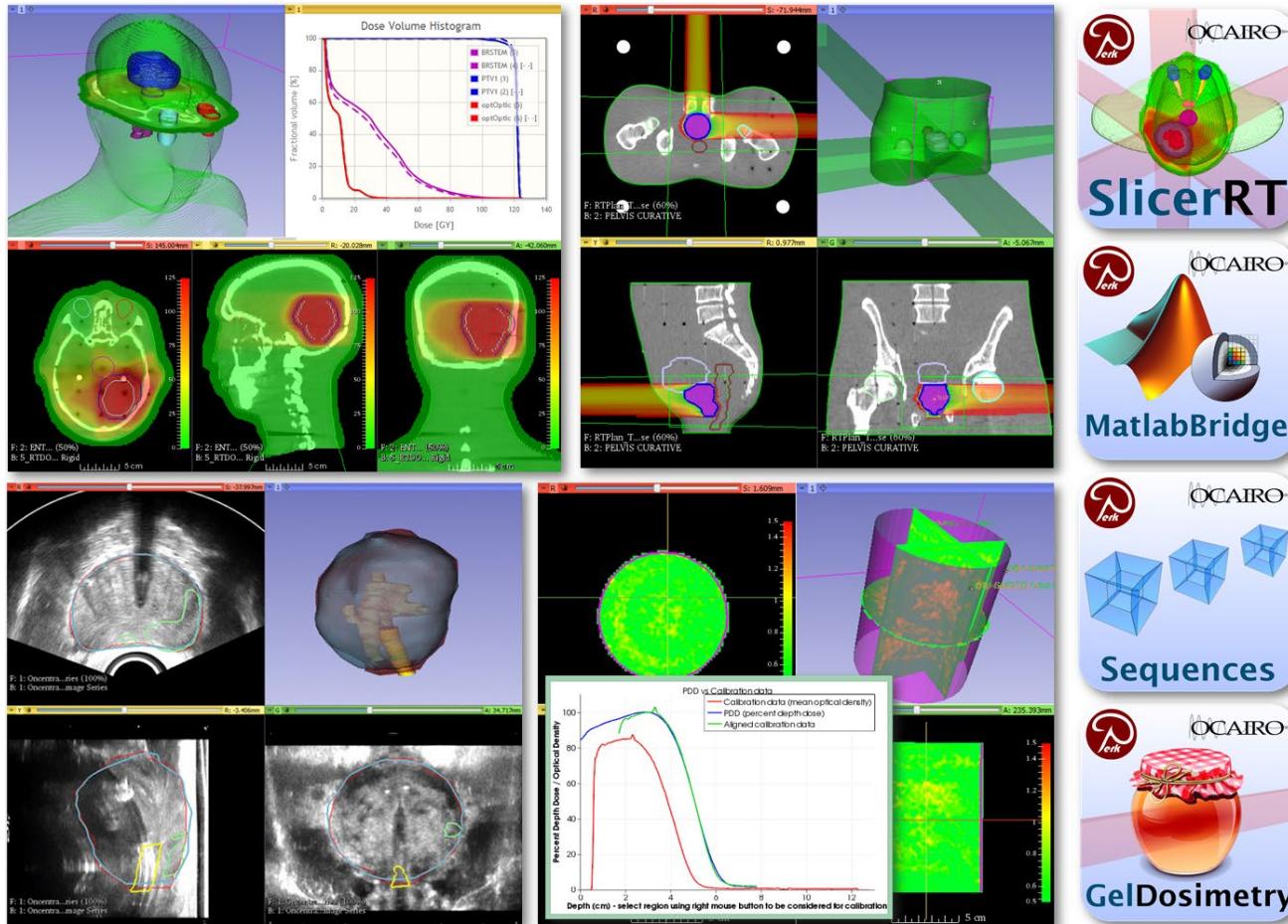
Extensão principal: SlicerIGT

- Toolkit para terapia guiada por imagem
- Base para navegação “guidelets”



Extensão principal: SlicerRT

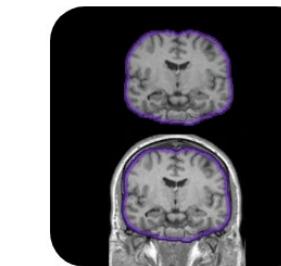
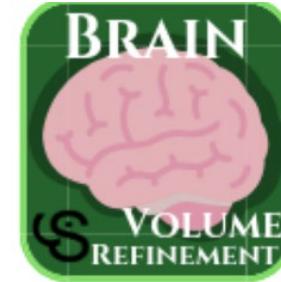
- Toolkit para radioterapia



Outras extensões notáveis



PerkTutor



SegmentEditorExtraEffects



DebuggingToolsForExtensions



Pacote de línguas
(LanguagePacks)

Tutoriais de introdução Slicer

Recomendado o uso do estável mais recente
3D Slicer Versão (Slicer-5.2.2) em todas as plataformas.

<https://download.slicer.org/>

Normalmente, após cerca de 3 meses, é melhor mudar para
a versão mais recente do Slicer Preview (devido às muitas
correções e melhorias disponíveis). O objetivo é criar uma
nova versão estável a cada 3 meses.

Seções de tutorial

- Visualização
 - carregar/salvar, dados de amostra, visualizadores, modelos, renderização de volume (30 min)
- DICOM
 - tags, onde obtê-los (web, navegador TCIA), opções de carregamento, plugins, exportação
 - Segmentação
 - Efeitos do Editor de segmentos, fluxos de trabalho
- Registration
 - BRAINS, Elastix, registro de ponto de referência, SegmentRegistration, transformação, transformar visualização
- Outro: Sequências, Jupyter, MatlabBridge, Realidade virtual

Reconstrução de volume a partir de Ultrassom segmentado por IA

Hu, Z., Nasute Fauerbach, P.V., Yeung, C. et al. Real-time automatic tumor segmentation for ultrasound-guided breast-conserving surgery navigation. Int J CARS (2022). doi: [10.1007/s11548-022-02658-4](https://doi.org/10.1007/s11548-022-02658-4)

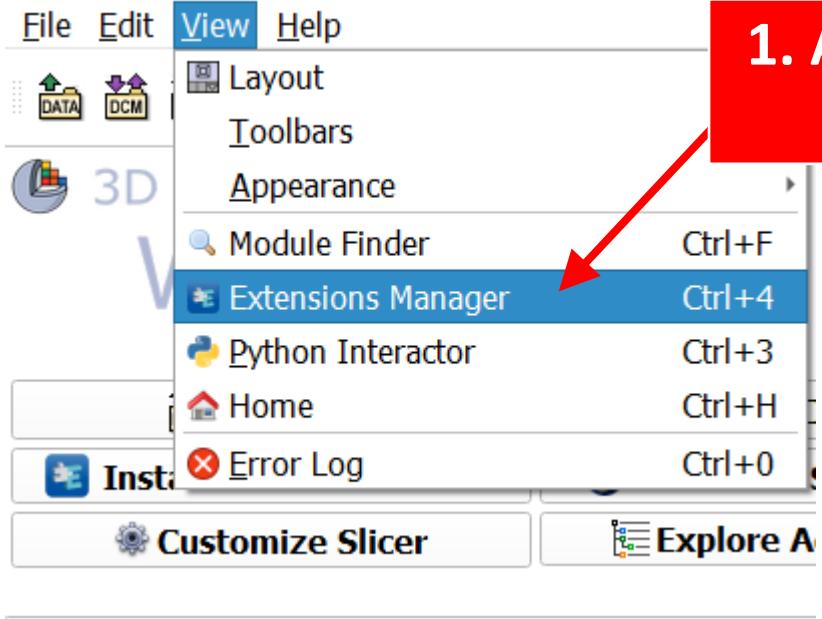
T. Ungi et al., "Automatic Spine Ultrasound Segmentation for Scoliosis Visualization and Measurement," in IEEE Transactions on Biomedical Engineering, vol. 67, no. 11, pp. 3234-3241, Nov. 2020, doi: [10.1109/TBME.2020.2980540](https://doi.org/10.1109/TBME.2020.2980540).

Pré-requisitos – baixar dados

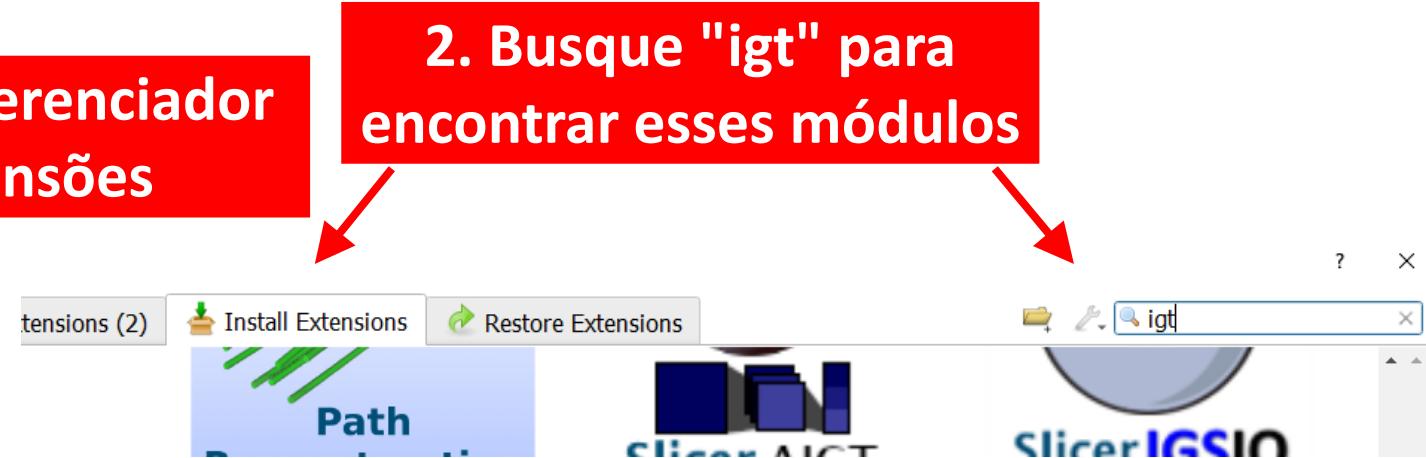
- Instalar a versão do 3D Slicer 5.0.2 ou posterior
- Baixar dados do tutorial (cena .mrb e Modelo de IA .h5)
 - Cena de dados: Q006_SagittalSpineScan.mrb (size: 389 MB)
<https://1drv.ms/u/s!AhiABcbe1DByhMt5E3NjCmJ-RIITZA?e=9Lqusg>
 - Modelo de IA treinado: SagittalSpine_05.h5 (size: 4 MB)
<https://1drv.ms/u/s!AhiABcbe1DByhMtwLY-z7kvKg1D5Ww?e=1loPhn>

Pré-requisitos – instalar extensões

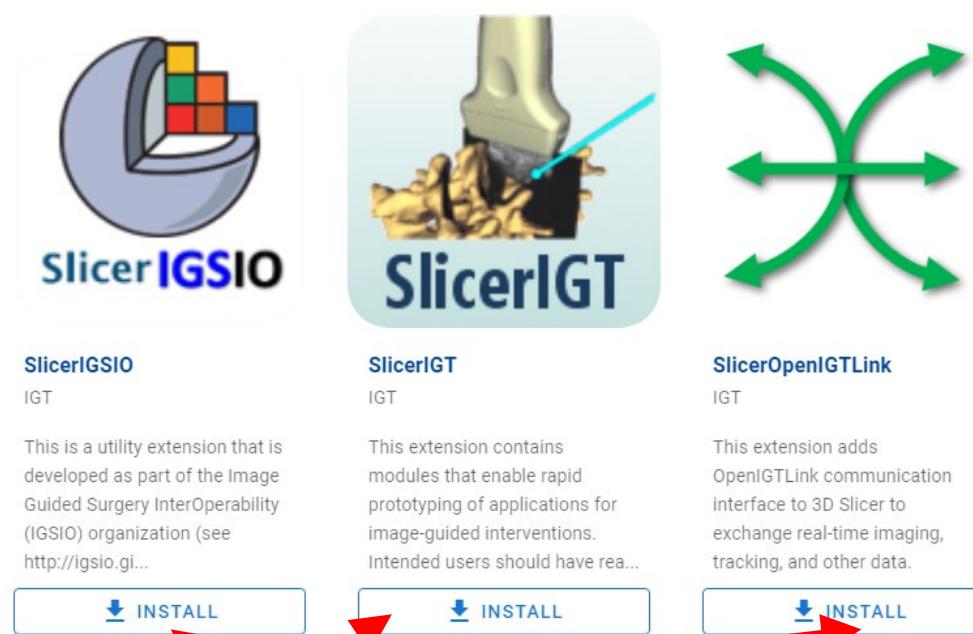
- Instalar extensões (plug-ins) para 3D Slicer
 - SlicerOpenIGTLink: Interface de comunicação com hardware
 - SlicerIGT: Módulos utilitários para IGT (Alinhamento e visualização)
 - SlicerIGSIO: Infraestrutura, módulo de algoritmos para IGT
 - ParallelProcessing: Modelo para executar scripts Python em paralelo



1. Abrir o Gerenciador de Extensões



2. Busque "igt" para encontrar esses módulos



3. Instalar

1. Instale a extensão ParallelProcesses também

2. Fechar, ainda não há necessidade de reiniciar a Segmentação de Dados

The screenshot shows the Extensions Manager window with the title bar 'Extensions Manager'. Below the title bar are buttons for 'Manage Extensions (3)', 'Install Extensions', and 'Restore Extensions'. A search bar contains the text 'parallel'. On the left, a sidebar titled 'Categories' lists 'All (1)' and 'Utilities (1)'. The main area displays a yellow icon with three interlocking gears and arrows, representing parallel processing. Below the icon, the extension is identified as 'ParallelProcessing' under the 'Utilities' category. A description states: 'Manage helper processes to perform background tasks in parallel.' A blue 'INSTALL' button is visible. At the bottom, a progress bar shows '0%' completion, and a note says '* Restart requested'. In the bottom right corner of the main window, there are 'Restart' and 'Close' buttons. A red arrow points from the text '2. Fechar...' to the 'Close' button.

Instale o TensorFlow em Slicer

- **Opcional!** Se você quiser que o TensorFlow use sua GPU, instale CUDA v. 11.3 e CuDNN v. 8.2
 - Para downloads e mais instruções, confira o site da NVidia:
<https://developer.nvidia.com/cuda-toolkit-archive>
- Em **Slicer / View/ Python Interactor** use o comando:
`>>> pip_install('tensorflow')`
- Quando o processo de instalação terminar, você poderá testar seu ambiente:
`>>> import tensorflow as tf`
`>>> tf.config.list_physical_devices()`



Clone o repositório SlicerIGT/aigt

- Clone repositório aigt no computador usando:
<https://github.com/SlicerIGT/aigt>
- Localizar *SegmentationUNet.py* no clone
...aigt/SlicerExtension/LiveUltrasoundAi/SegmentationUNet/
- Abrir **Slicer / Edit / Application Settings / Modules**
- Solte o arquivo SegmentationUNet.py na área em Slicer configurações chamadas *Additional module caminhos*
- Pressione OK na janela Configurações e reinicie o Slicer

File Edit View Help

Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V

Record Macro
Play Macro

Application Settings Ctrl+2

Add Data

DCM Add D

Slicer

File Home Share View

Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut

Move to
Copy to
Delete
Organize

R

2. Selecionar Modules

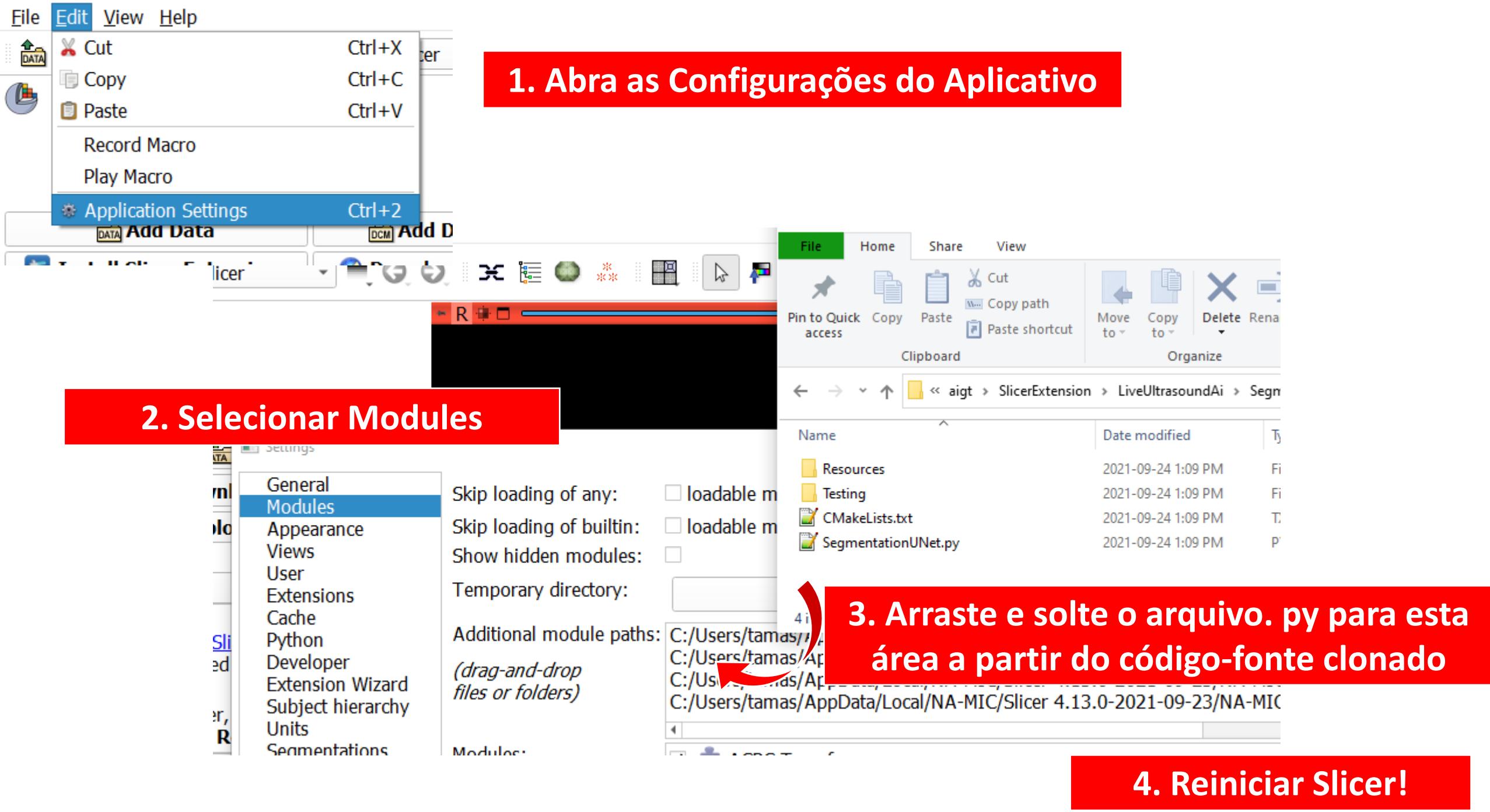
General Modules

Appearance
Views
User
Extensions
Cache
Python
Developer
Extension Wizard
Subject hierarchy
Units
Segmentations

Skip loading of any: loadable modules
Skip loading of builtin: loadable modules
Show hidden modules:
Temporary directory:
Additional module paths:
(drag-and-drop files or folders)
Modules:

3. Arraste e solte o arquivo. py para esta área a partir do código-fonte clonado

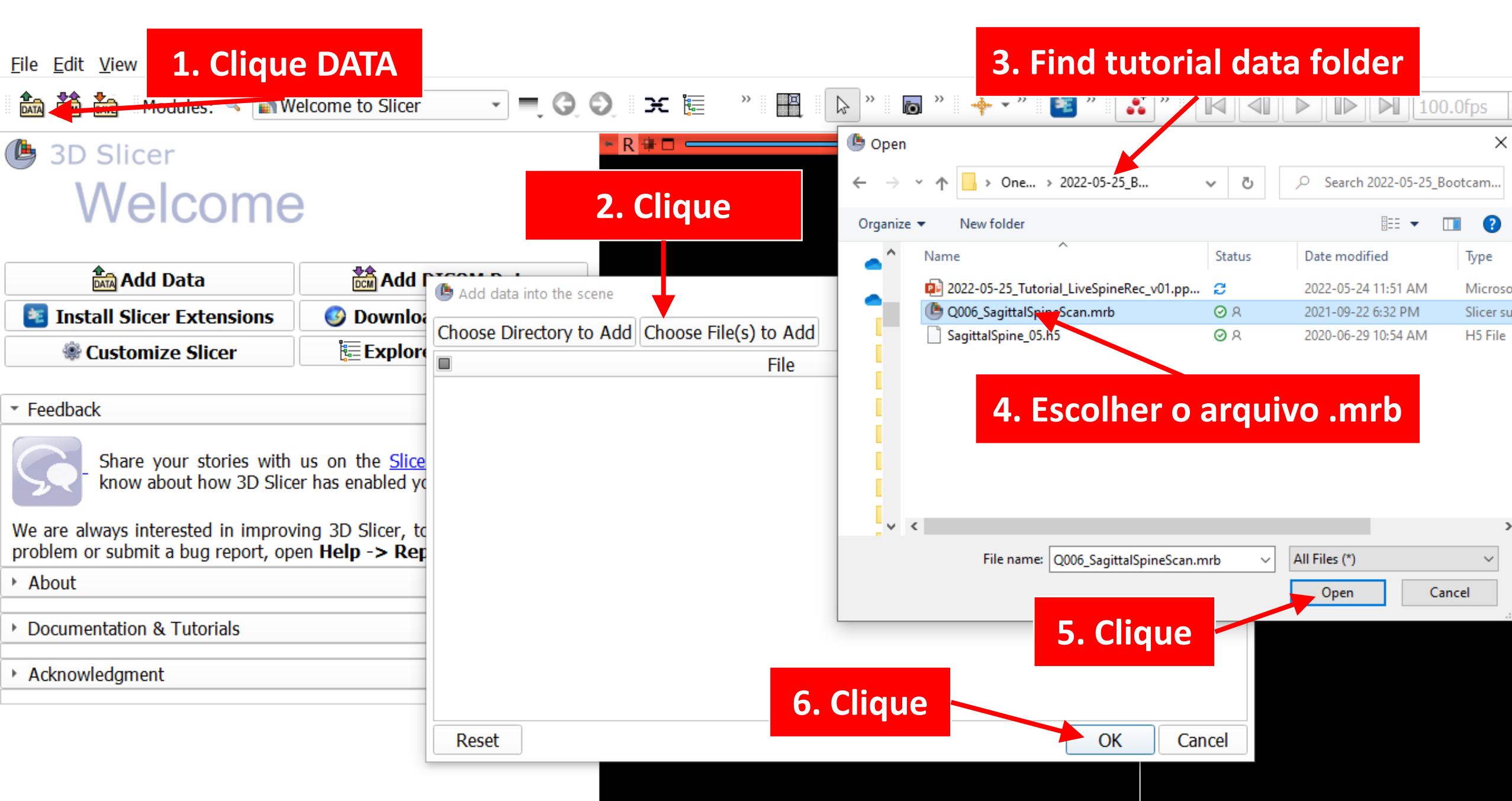
4. Reiniciar Slicer!



The image shows a composite screenshot of the Slicer application. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Help'. Below it is a toolbar with various icons. A red box highlights the 'File' menu. On the left, a vertical sidebar lists 'General', 'Modules', 'Appearance', 'Views', 'User', 'Extensions', 'Cache', 'Python', 'Developer', 'Extension Wizard', 'Subject hierarchy', 'Units', and 'Segmentations'. A red box highlights the 'General' and 'Modules' items. In the center, there's a configuration dialog for 'General Modules'. It has sections for skipping loading of any or builtin modules, showing hidden modules, setting a temporary directory, and adding additional module paths via a drag-and-drop area. A red arrow points to this area. To the right, a file explorer window shows a folder structure under 'LiveUltrasoundAi'. A red box highlights the 'Segmentations' folder. Below the file explorer is a terminal window with some text. A large red box covers the bottom right corner of the image.

Visão geral do processamento

- Carregar cena da segmentação de dados
- Configurar entrada e saída para o módulo de segmentação
- Inicializar a previsão atualizando a entrada
- Visualizar previsão
- Configurar a reconstrução de volume
- Iniciar repetição de ultrassom
- Configurar a renderização de volume para visualizar o volume da coluna



3D Slicer 4.13.0-2021-09-23

File Edit View Help



3D Slicer

Welcome

Load DICOM Data

Install Slicer Extensions

Customize Slicer

Feedback



Share your stories
about how 3D Slicer

We are always interested in improving 3D Slicer or submit a bug report, open an issue, or contribute to the code.

About

Documentation & Tutorials

Acknowledgment

Welcome to Slicer

- Annotations
- Data
- DataStore
- DICOM
- Markups
- Models
- Scene Views
- Segment Editor
- Segmentations
- Transforms
- View Controllers
- Volume Rendering
- Volumes
- Welcome to Slicer

Ultrasound

Wizards

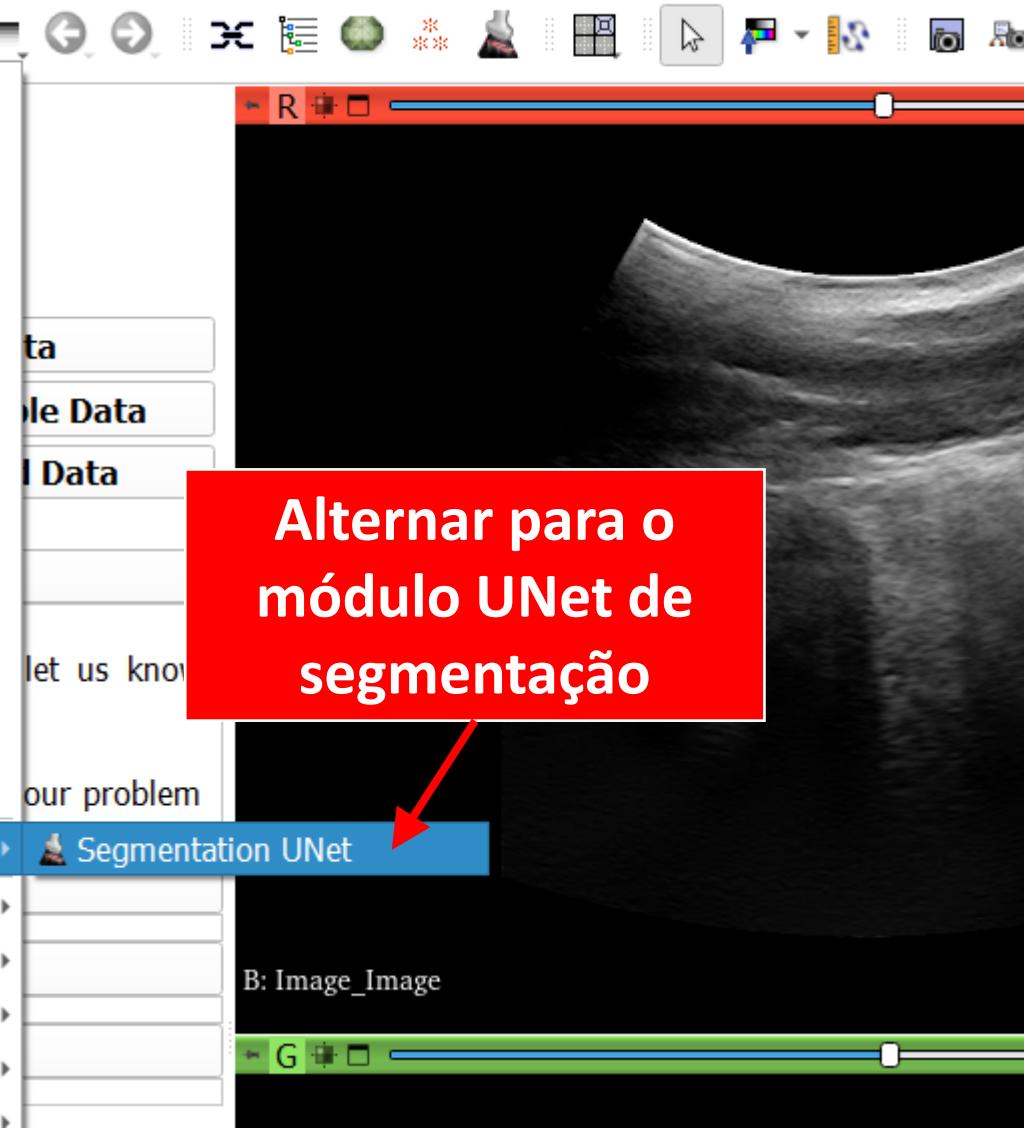
Informatics

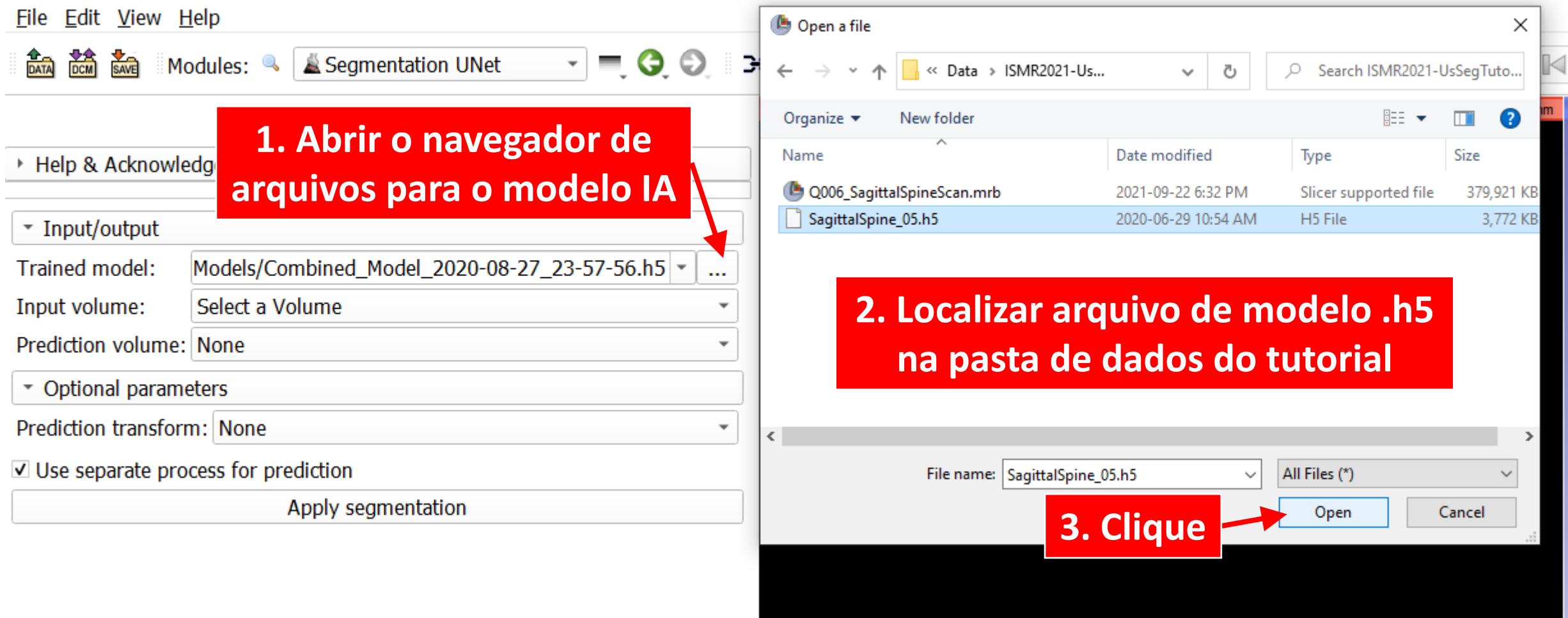
Registration

Segmentation

Quantification

Alternar para o
módulo UNet de
segmentação





1. Selecionar Image_Image como volume de entrada

Input/output

Trained model: C:/Users/tamas/Downloads/SagittalSpine_05.h5

Input volume: Image_Image

Prediction volume: None

Optional parameters

Prediction transform: None

Use separate process

None
Image_Image
Rename current Volume
Create new Volume
Create new Volume as...
Delete current Volume

2. Criar novo volume para Previsão

Rename Vol... ? X

New name:
Prediction

OK Cancel

1. Criar novo nó de transformação para Previsão

Optional parameters

Prediction transform: None

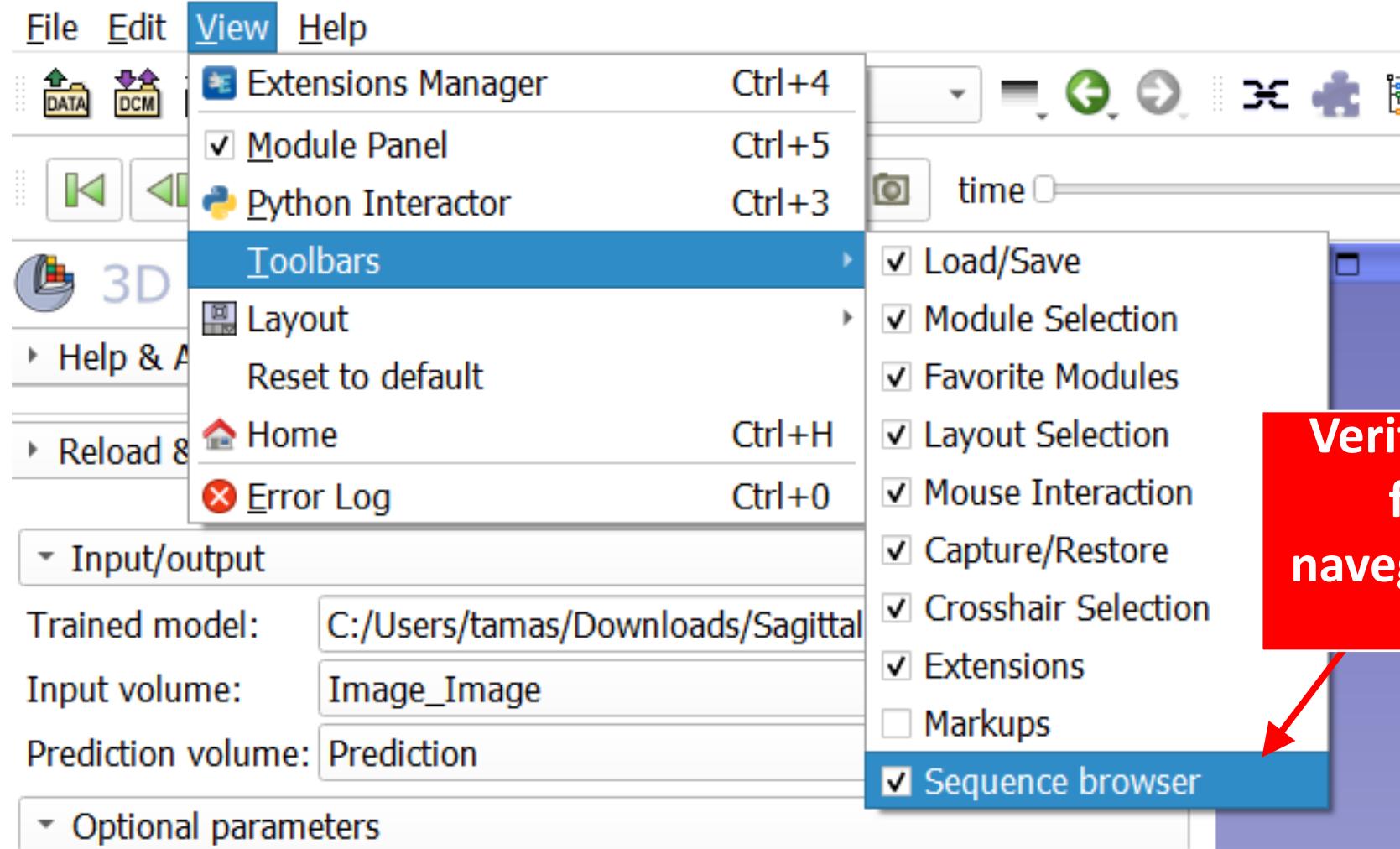
Use separate process

None
ImageToReference
StylusToReference
ReferenceToRas
Rename current LinearTransform
Create new LinearTransform
Create new LinearTransform as...
Delete current LinearTransform

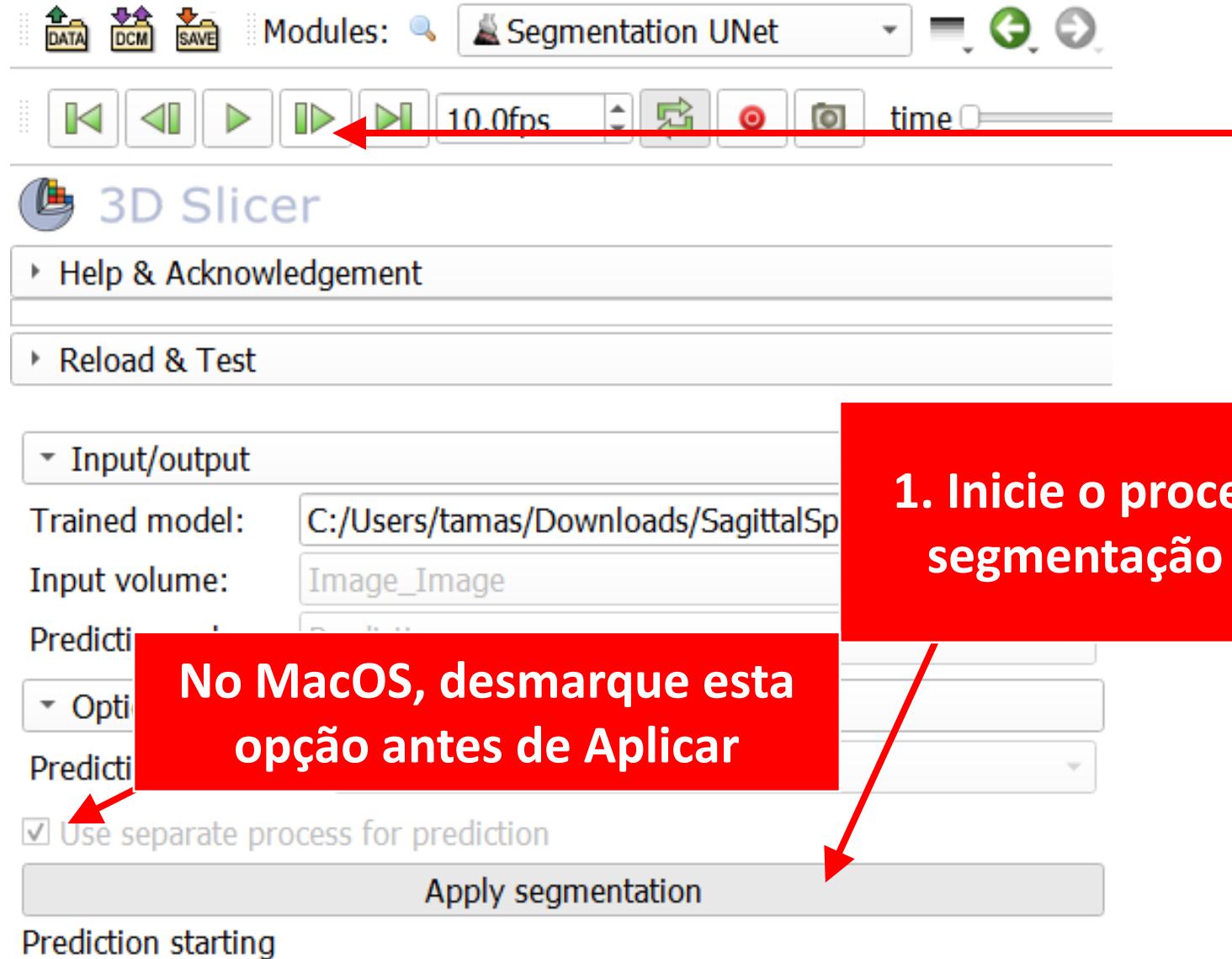
Rename Line... ? X

New name:
PredictionToRas

OK Cancel

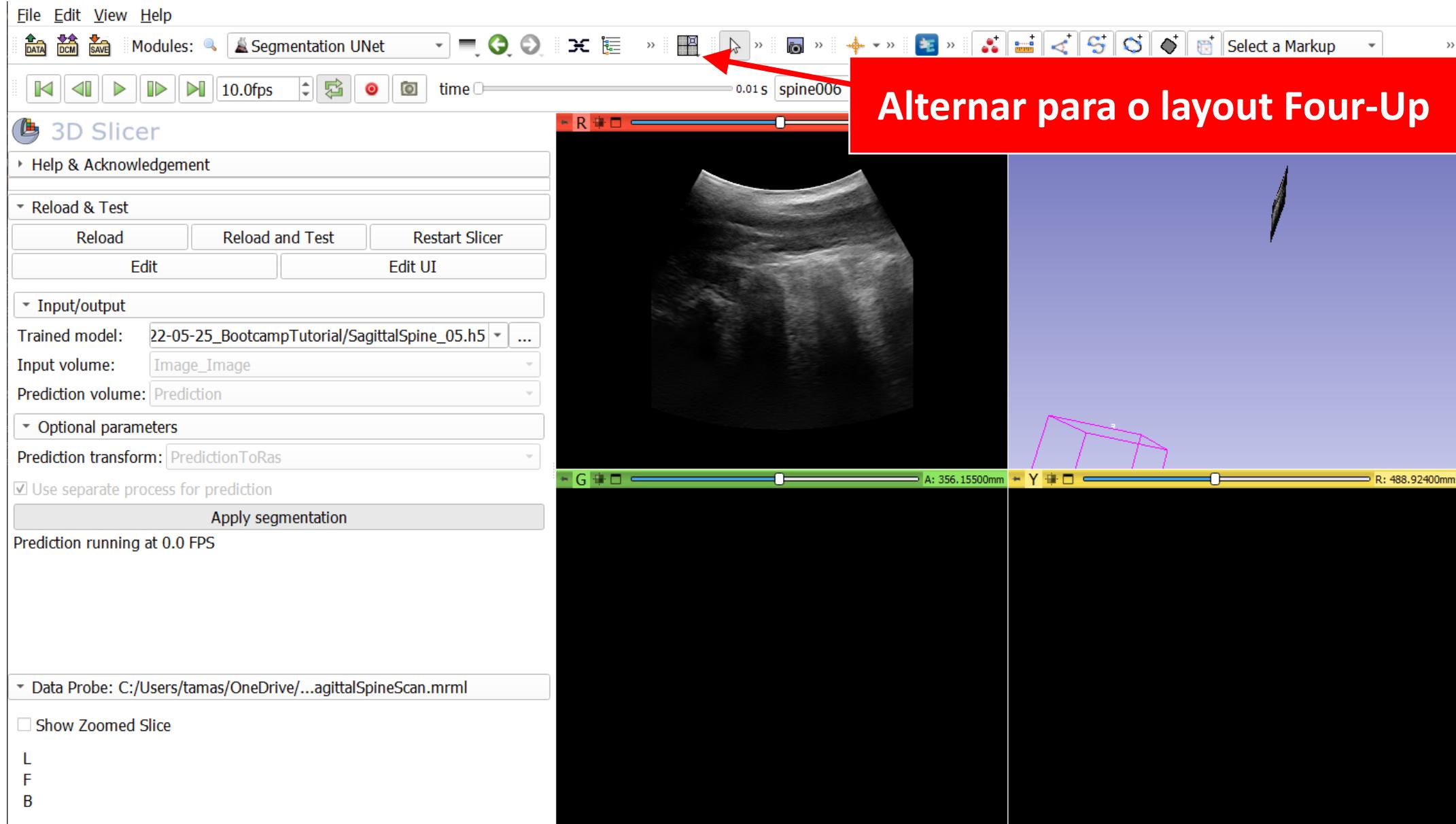


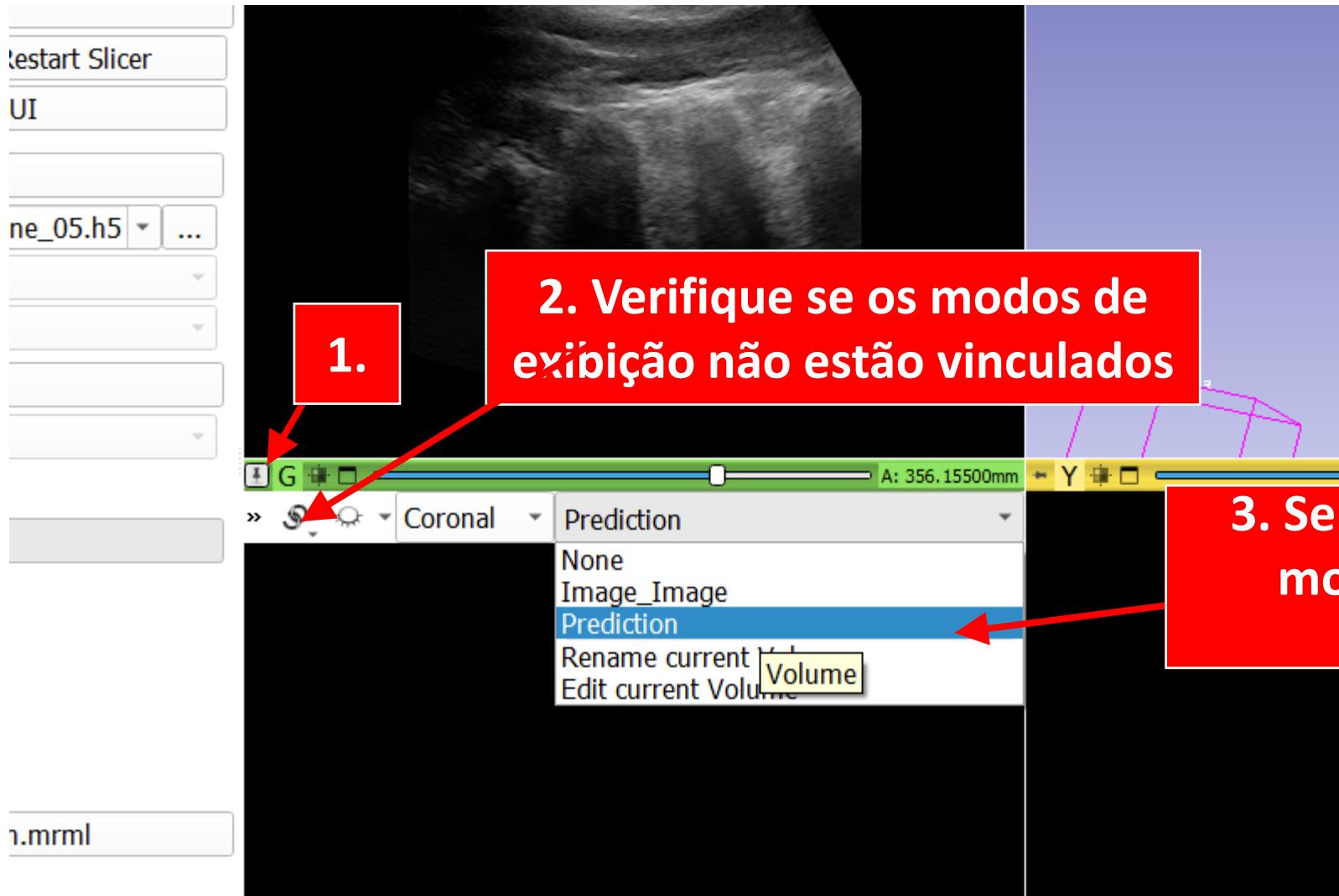
Verifique se a barra de ferramentas do navegador de sequência está visível

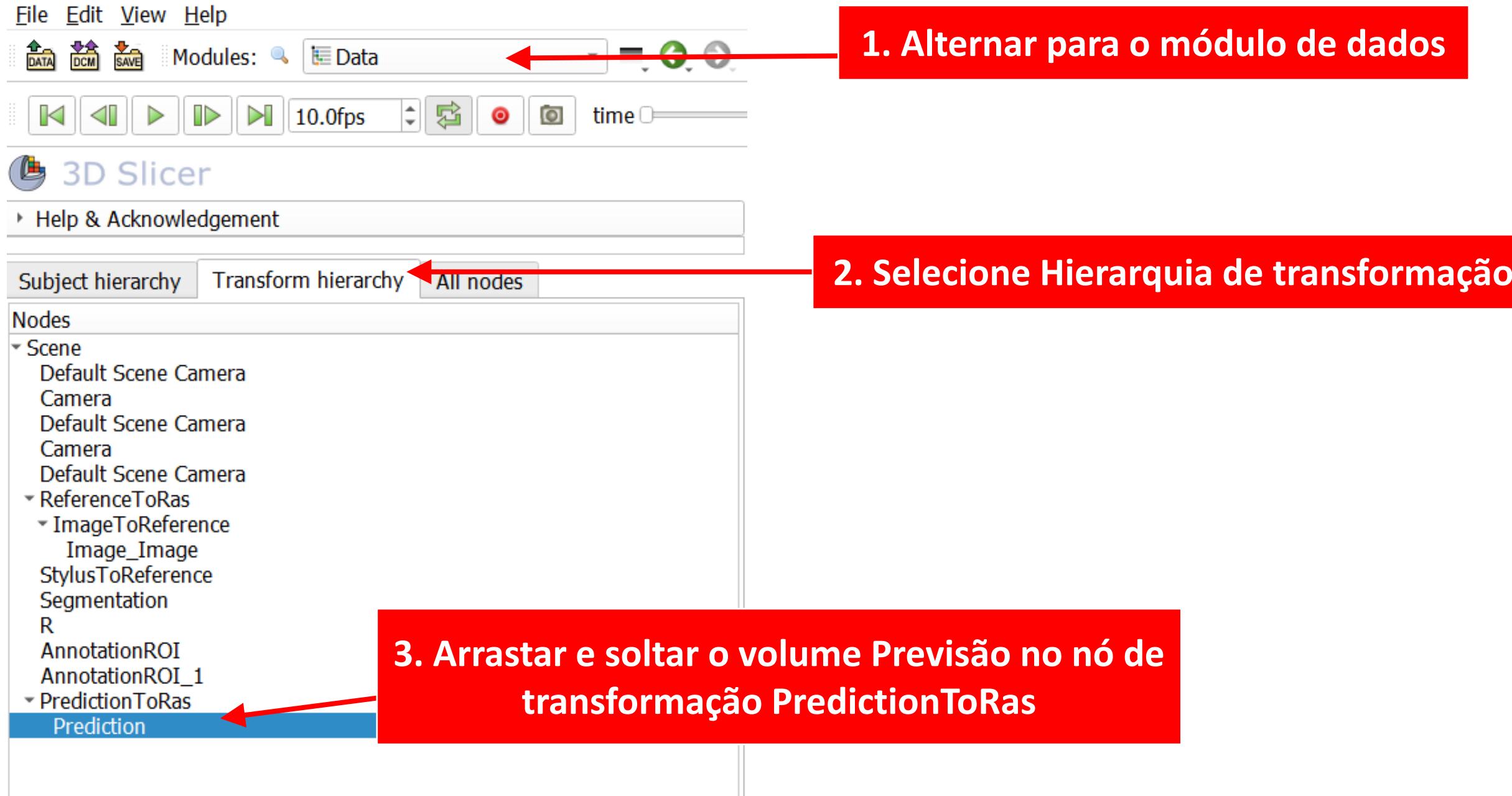


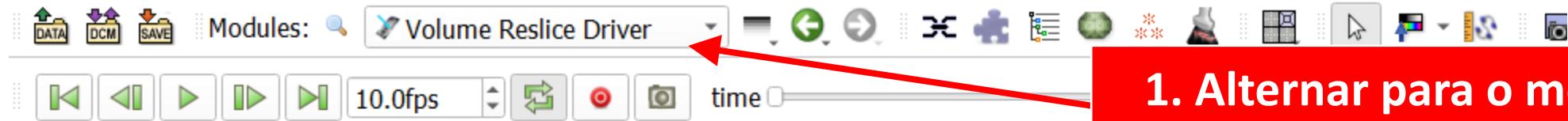
1. Inicie o processo de segmentação de AI

2. Siga a sequência gravada para que o processo de AI seja totalmente iniciado









3D Slicer

Help & Acknowledgement

Reslicing Driver

R □

Driver: Image_Image

Mode: Transverse

Rotation: 0

G □

Driver: Prediction

Mode: Transverse

Rotation: 0

Flip

Y □

Driver: None

Mode: Off

Rotation: 0

Flip

Advanced options

Show advanced options

1. Alternar para o módulo Volume Reslice Drive (categoria IGT)

2. Selecione Previsão para conduzir a visualização verde

3. Mostrar opção avançada e inverter a imagem de previsão para que ela corresponda à orientação do ultrassom



3D Slicer

Help & Acknowledgement

Volume reconstruction

Volume reconstruction node: VolumeReconstruction

Input method: Recorded sequence reconstruction
 Live volume reconstruction

Input sequence browser: Select a SequenceBrowser

Input volume node: Prediction

Live update interval: 1.00s

Output volume node: ReconstructedVolume

ROI node: R

Output spacing: 1.00 1.00 1.00

Interpolation mode: Linear

Optimization mode: Full optimization

Compounding mode: Maximum

Fill holes:

Number of threads: 0

Skip interval: 0

Clipping

Clip rectangle origin: 0 0

Clip rectangle size: 0 0

Reset

Start

1. Mudar para o módulo de Reconstrução de Volume (categoria IGT)

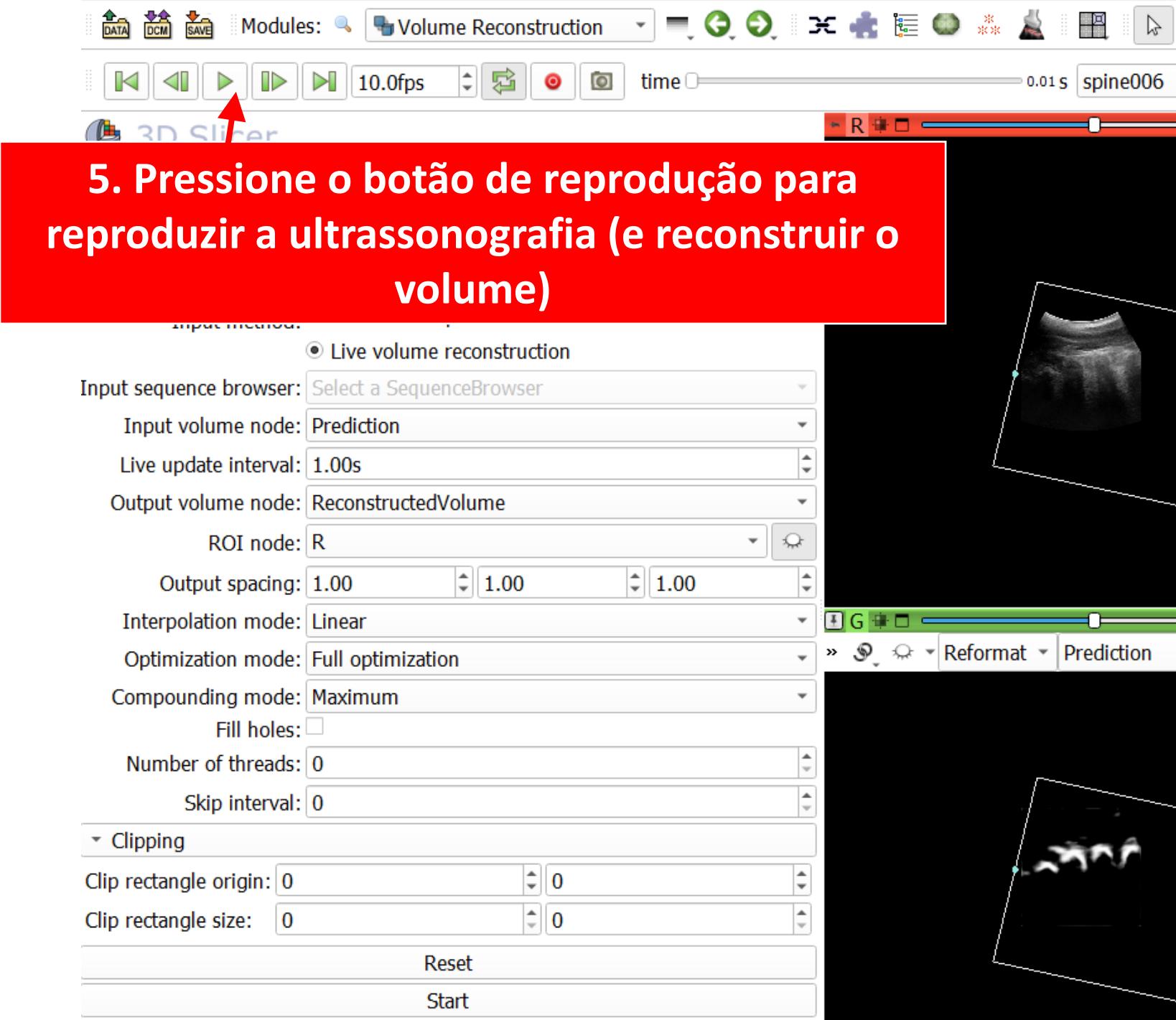


2. Select

2. Selecione estas opções

3. Criar um novo volume e nomeá-lo ReconstructedVolume para saída

4. Pressione Iniciar







1. Alternar para o modo de exibição somente 3D

2. Selecionar Anotações

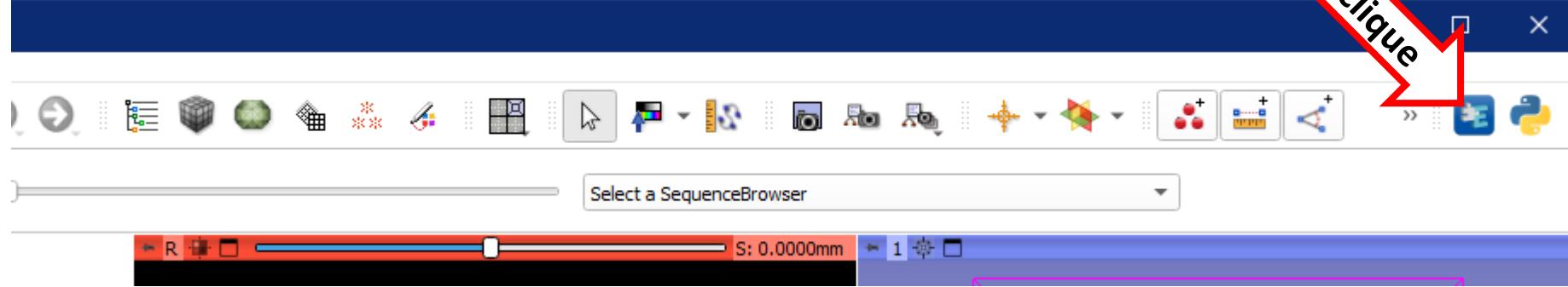
3. Ocultar R (ROI)

4. Parar a reprodução quando o volume estiver pronto

(5. Parar a reconstrução de volume e segmentação módulos UNet também)

The main workspace displays a 3D volume rendering of a spine. On the left, the 'Annotations' module panel is open, showing a list of ROI objects: 'ROI List', 'AnnotationROI_2', 'R' (which is currently selected and highlighted in blue), and 'AnnotationROI'. A red arrow points to the 'R' entry in the list. The bottom section contains descriptive text boxes with numbered steps corresponding to the interface elements.

Instalar extensões Slicer



2. Find and install these extensions:
SlicerIGT, SlicerDMRI

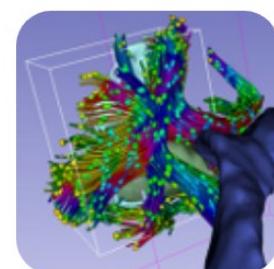


SlicerIGT

IGT

This extension contains modules that enable rapid prototyping of applications for guided interventions. End users should have rea...

3. clique
[INSTALL](#)

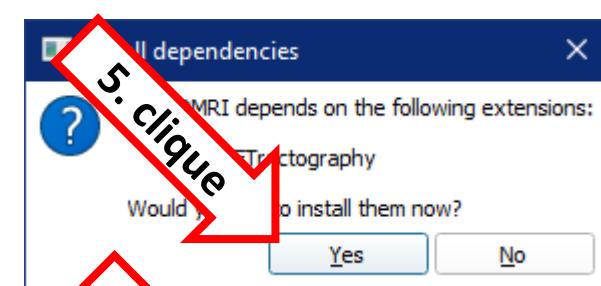


SlicerDMRI

Diffusion

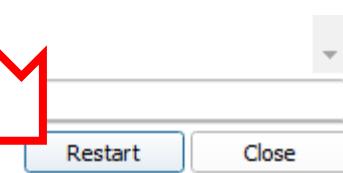
SlicerDMRI is an umbrella extension for Slicer Diffusion MRI functionality.

4. clique
[INSTALL](#)

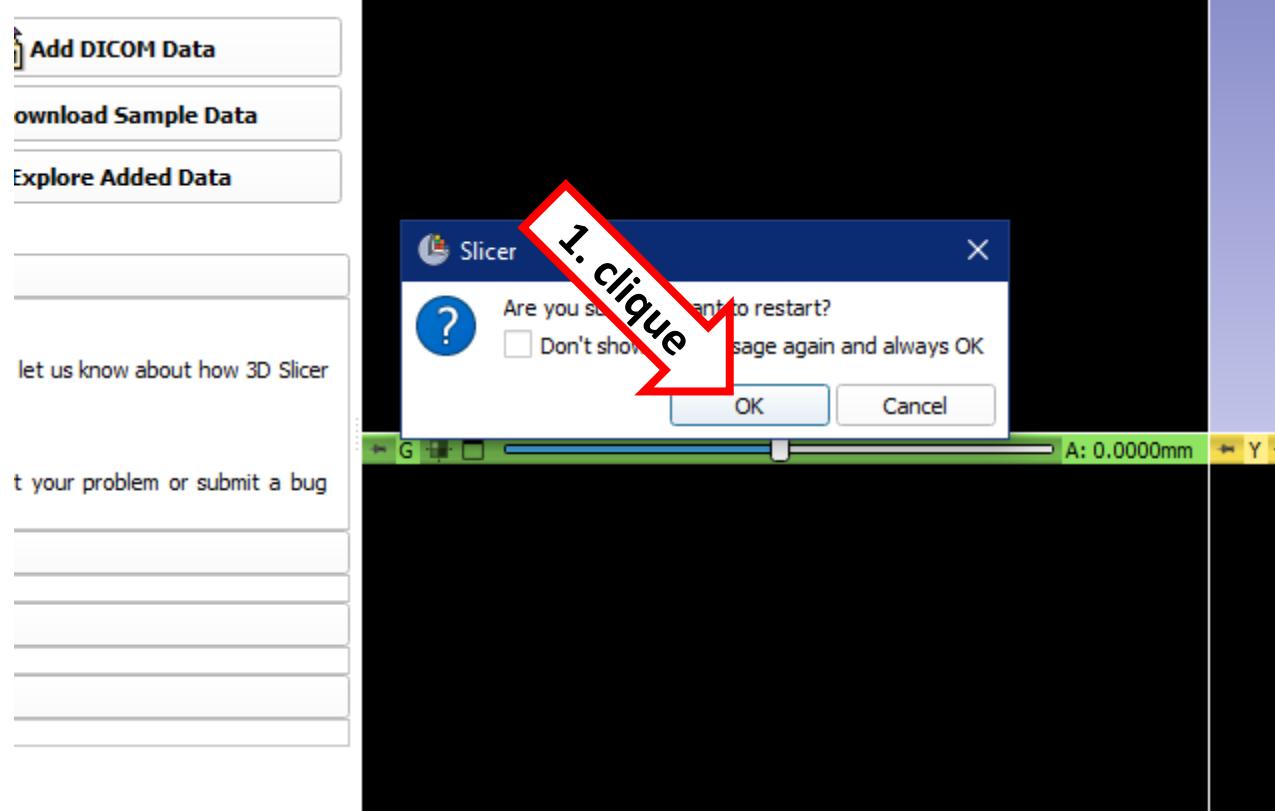


5. clique

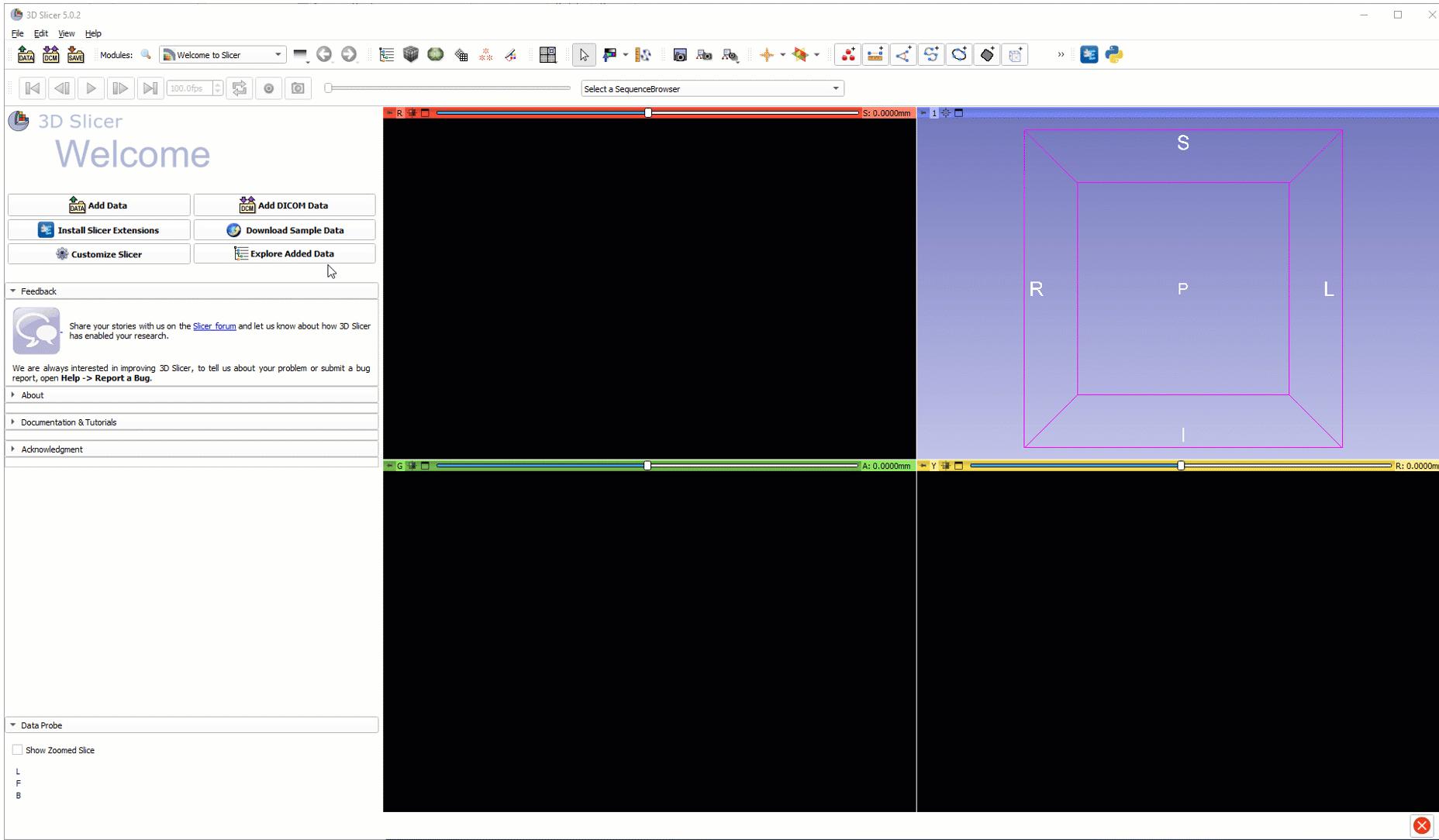
6. clique



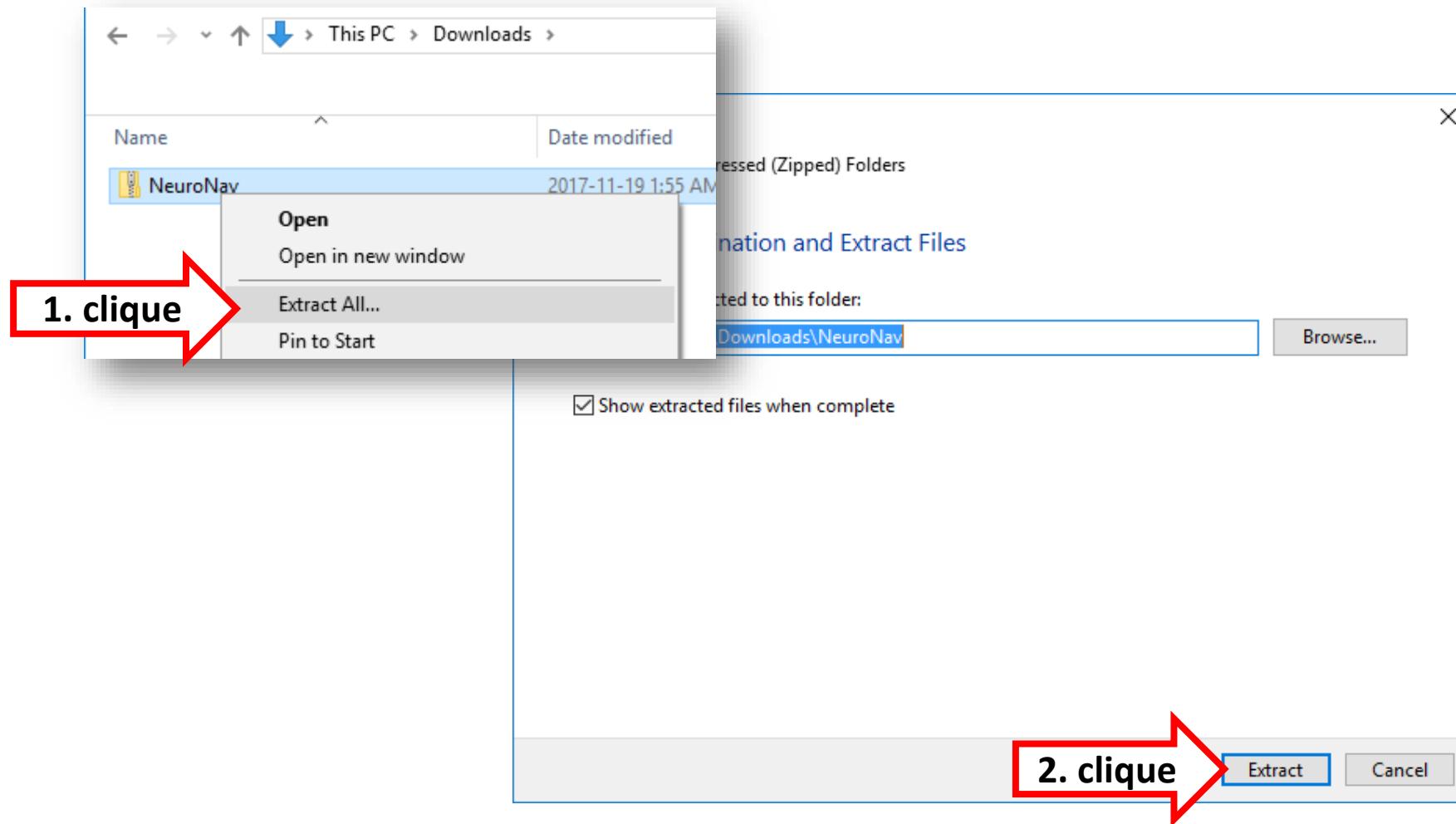
Aprovar a reinicialização da Segmentação de Dados (se solicitado)

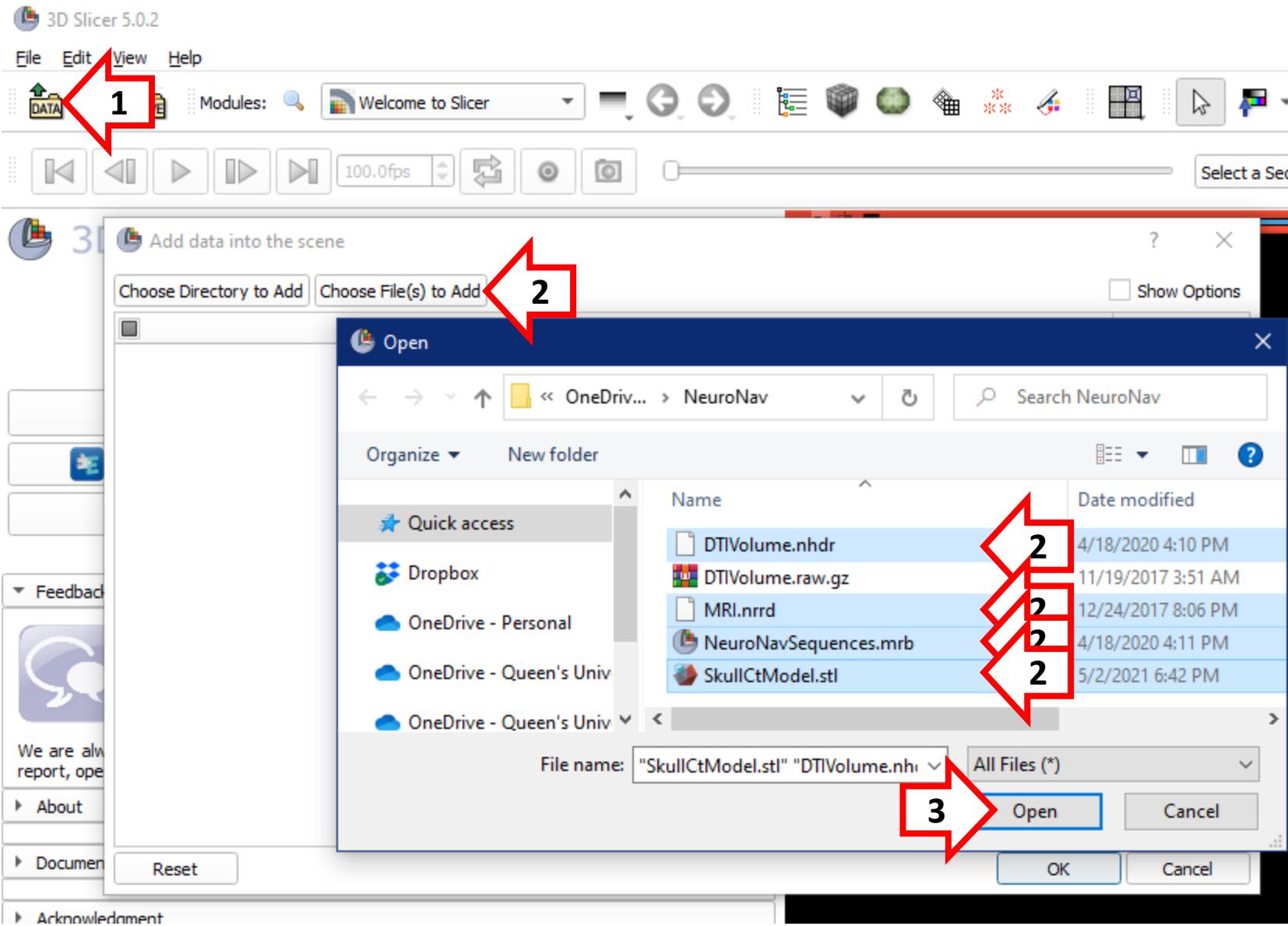


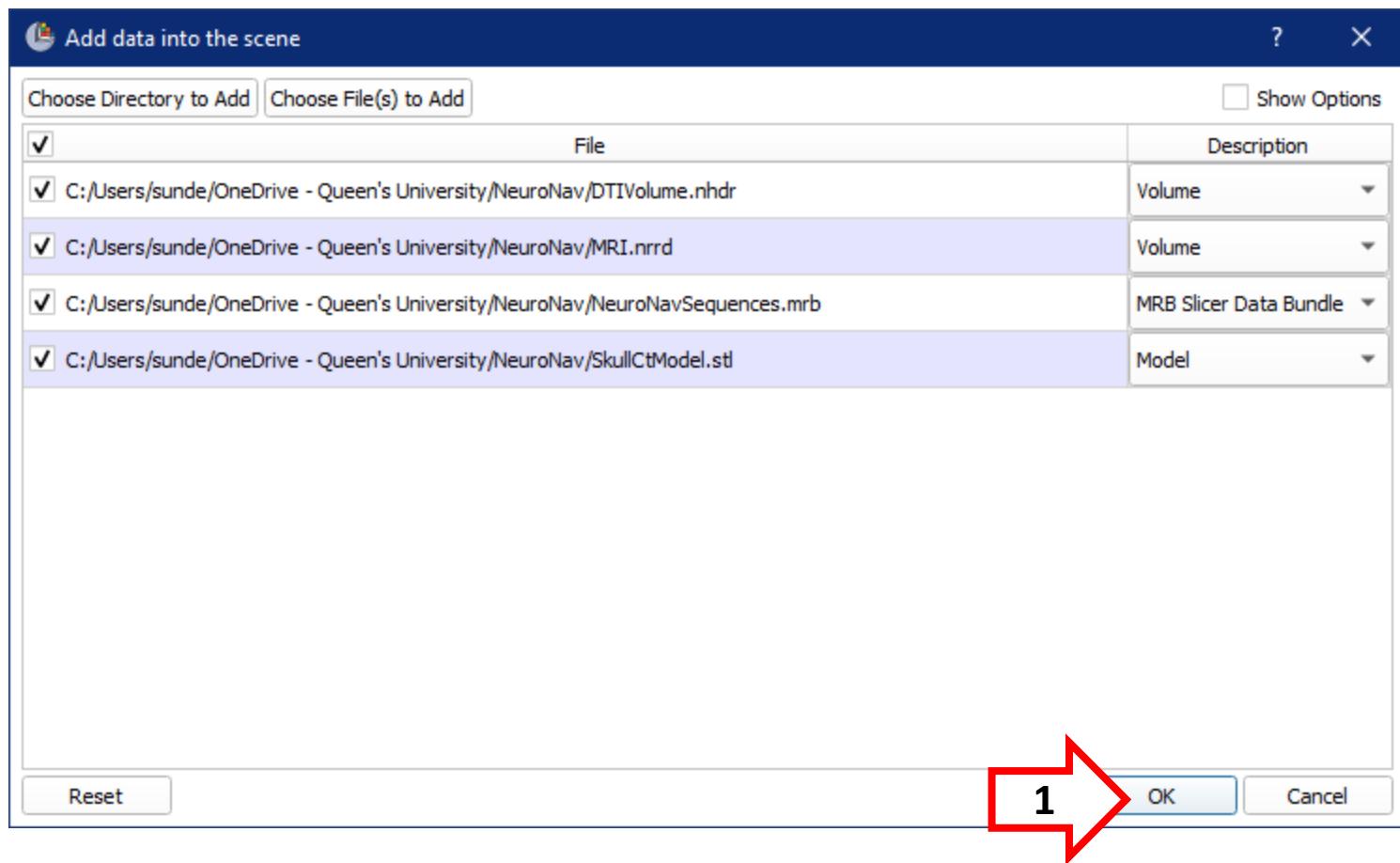
Verificar extensões instaladas



Unzip a pasta de dados do tutorial

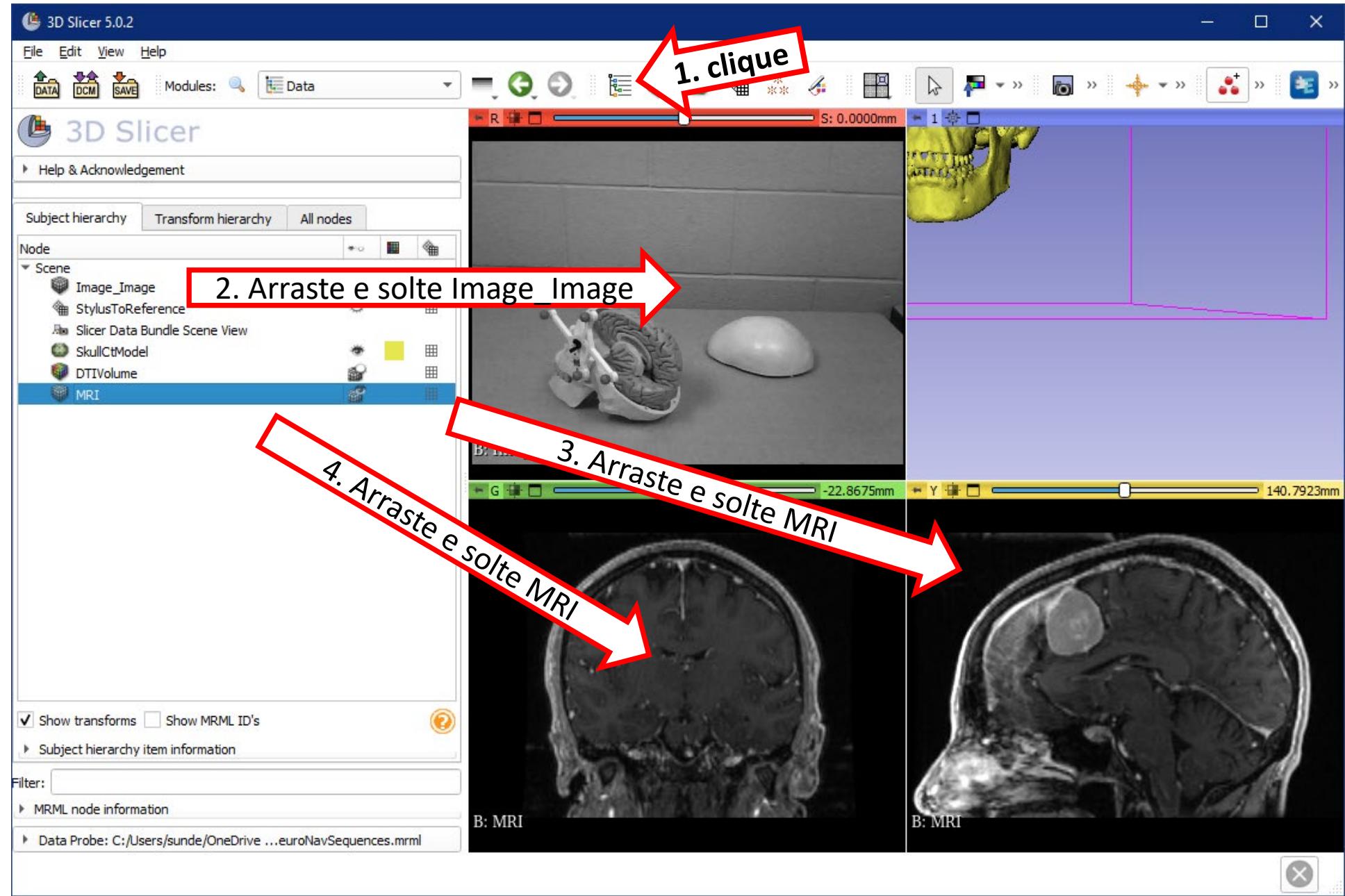


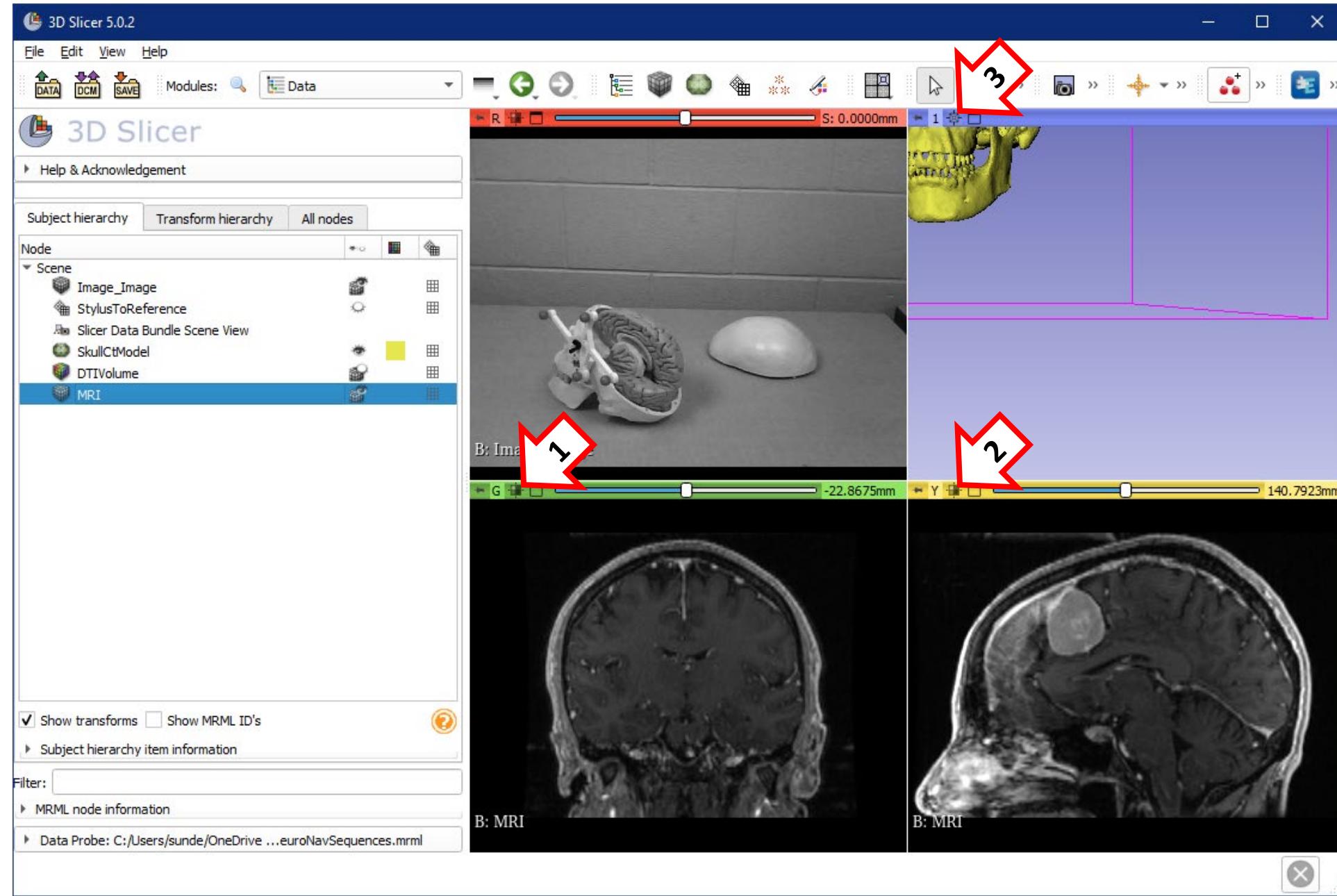


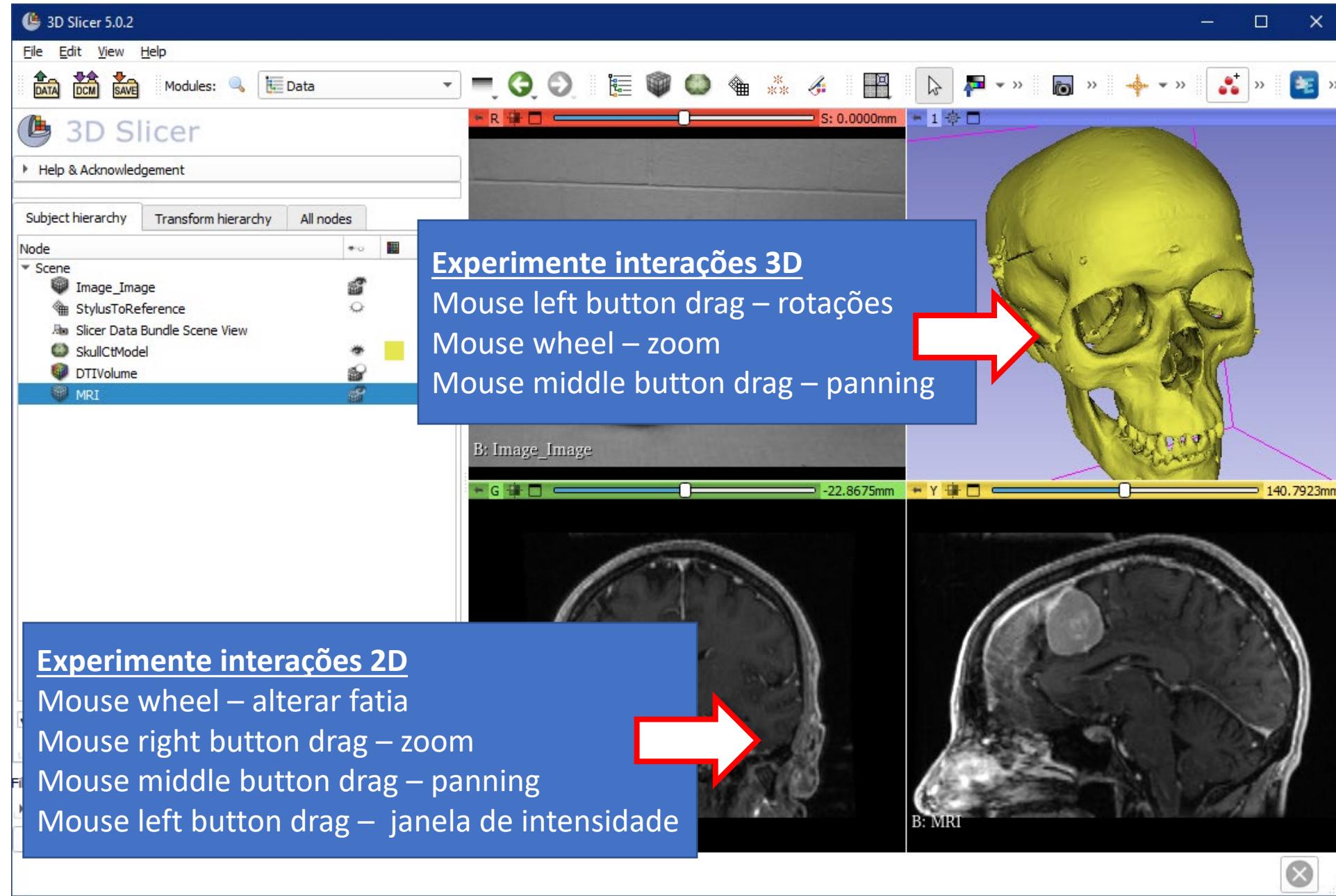


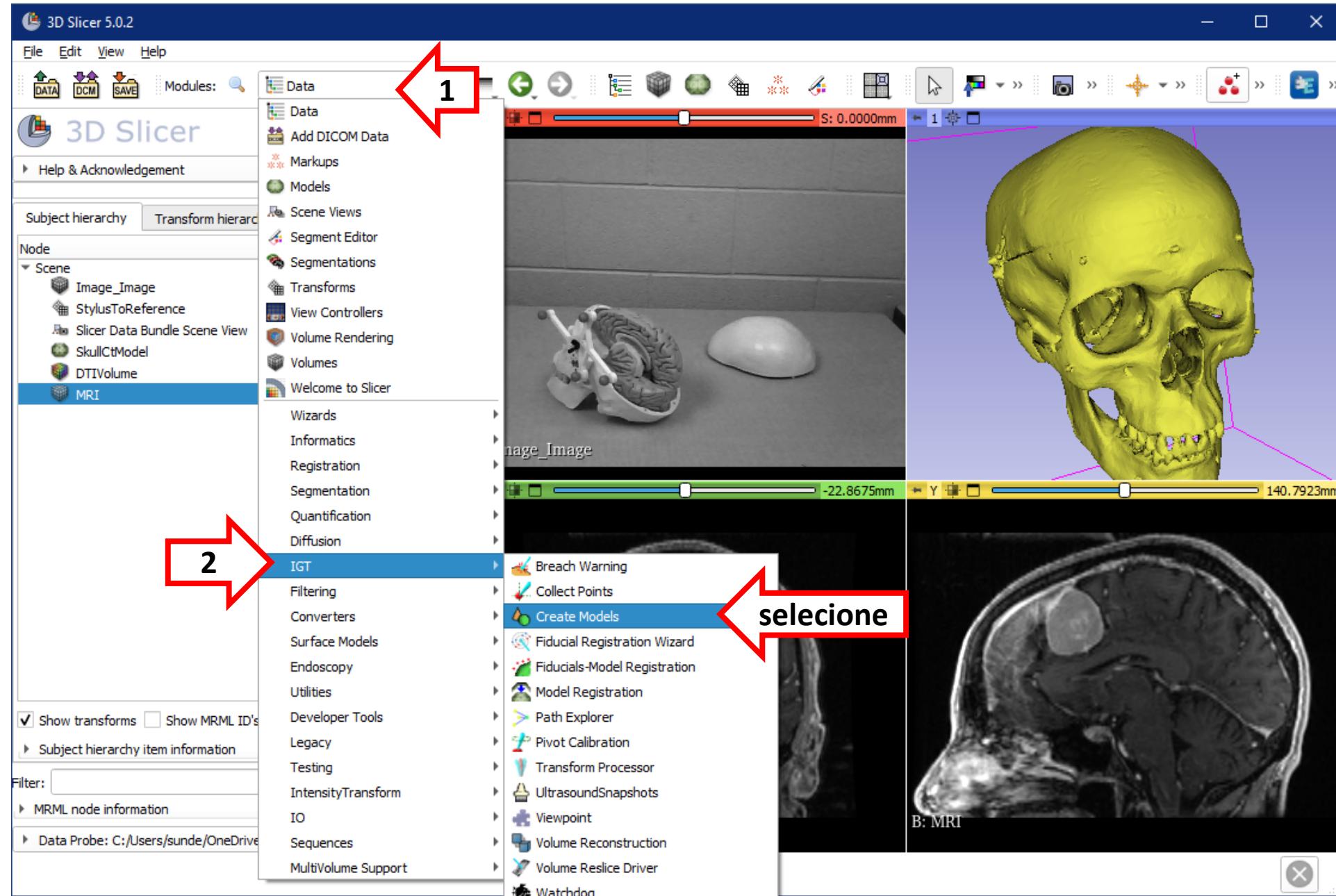
Sobre os dados carregados

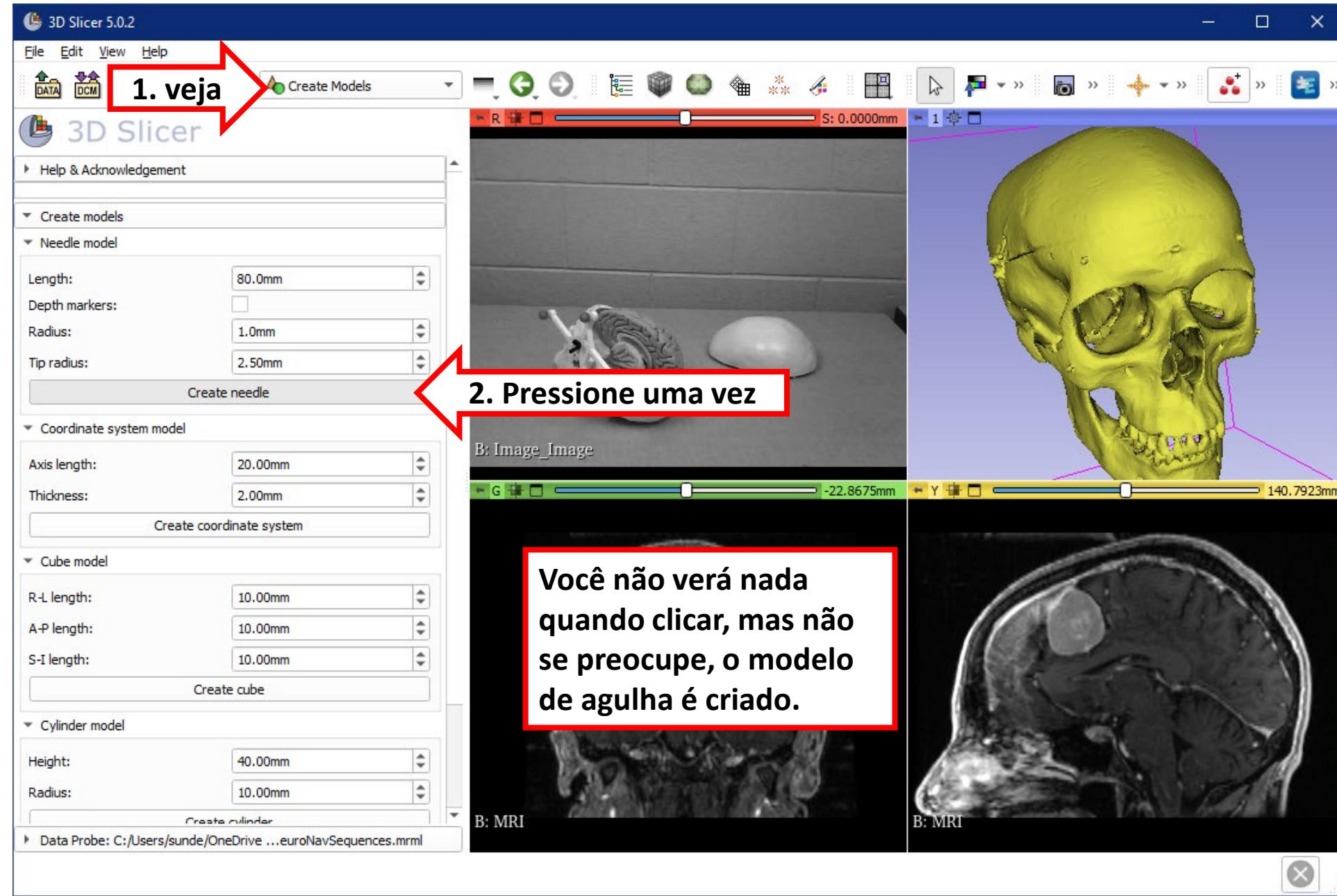
- **SkullCtModel.stl**: Segmentação a partir da tomografia computadorizada de um modelo de crânio de plástico
- **MR1.nrrd**: Imagem de ressonância magnética de uma pessoa. (não o mesmo em que o modelo de plástico foi baseado)
- **DTIVolume.nhdr**: Imagem do tensor de difusão (de uma terceira pessoa)
- **NeuroNavSequences.mrb**: Sequências de tempo para reproduzir movimentos de ferramentas
-
- Todos os volumes de imagem já estão registrados aproximadamente (eles mais ou menos se sobrepõem, mas vêm de pacientes diferentes, então a sobreposição não é perfeita)

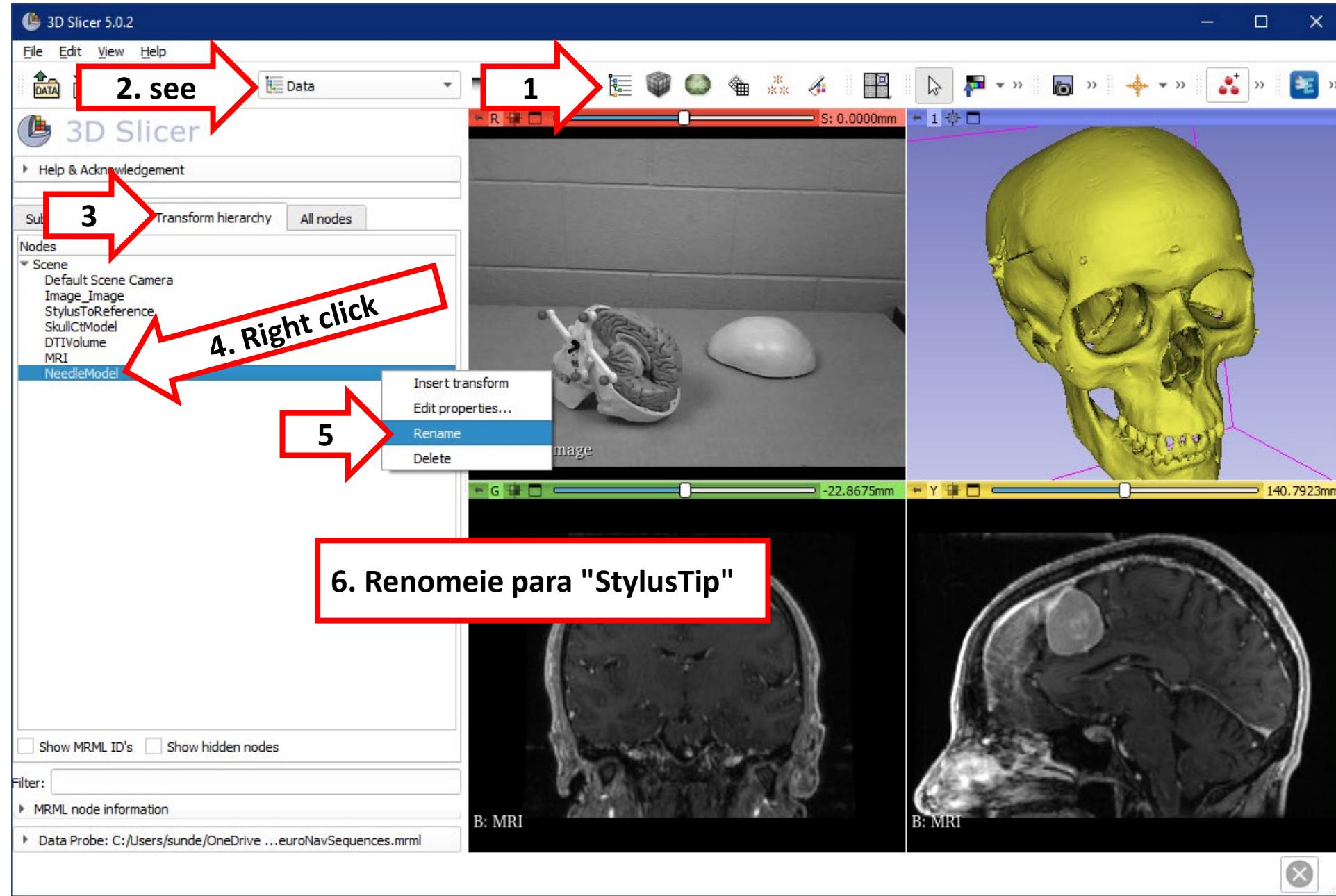


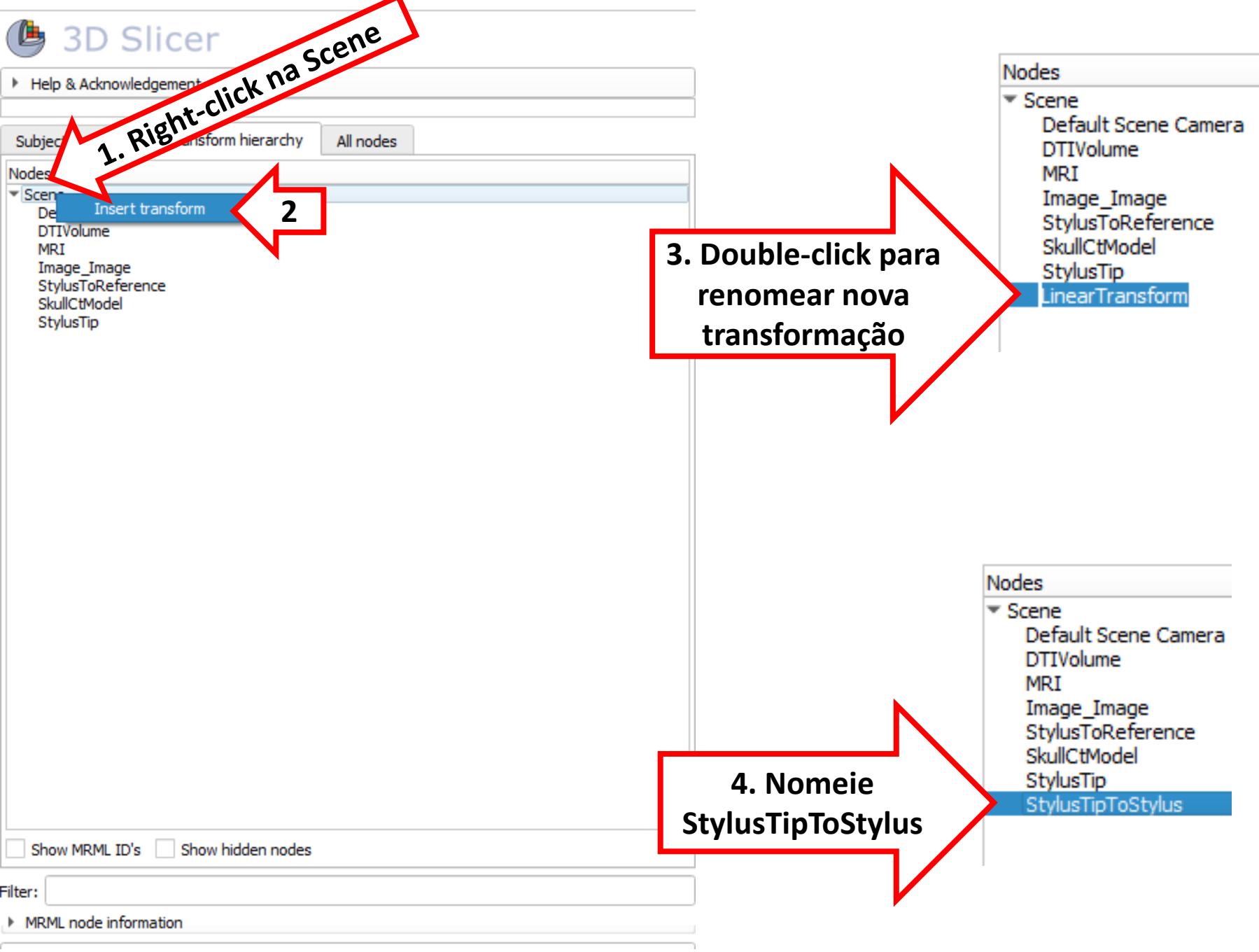


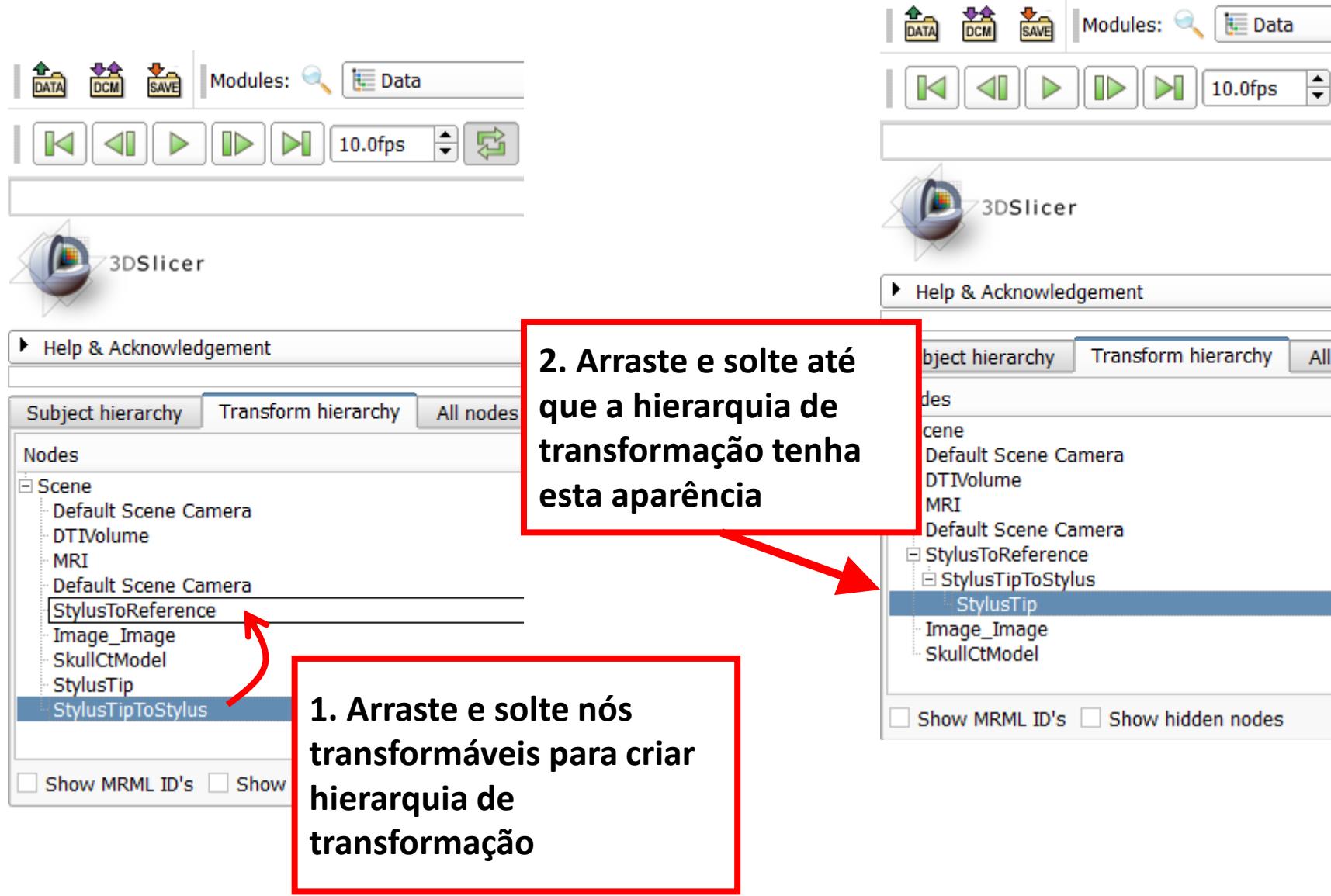


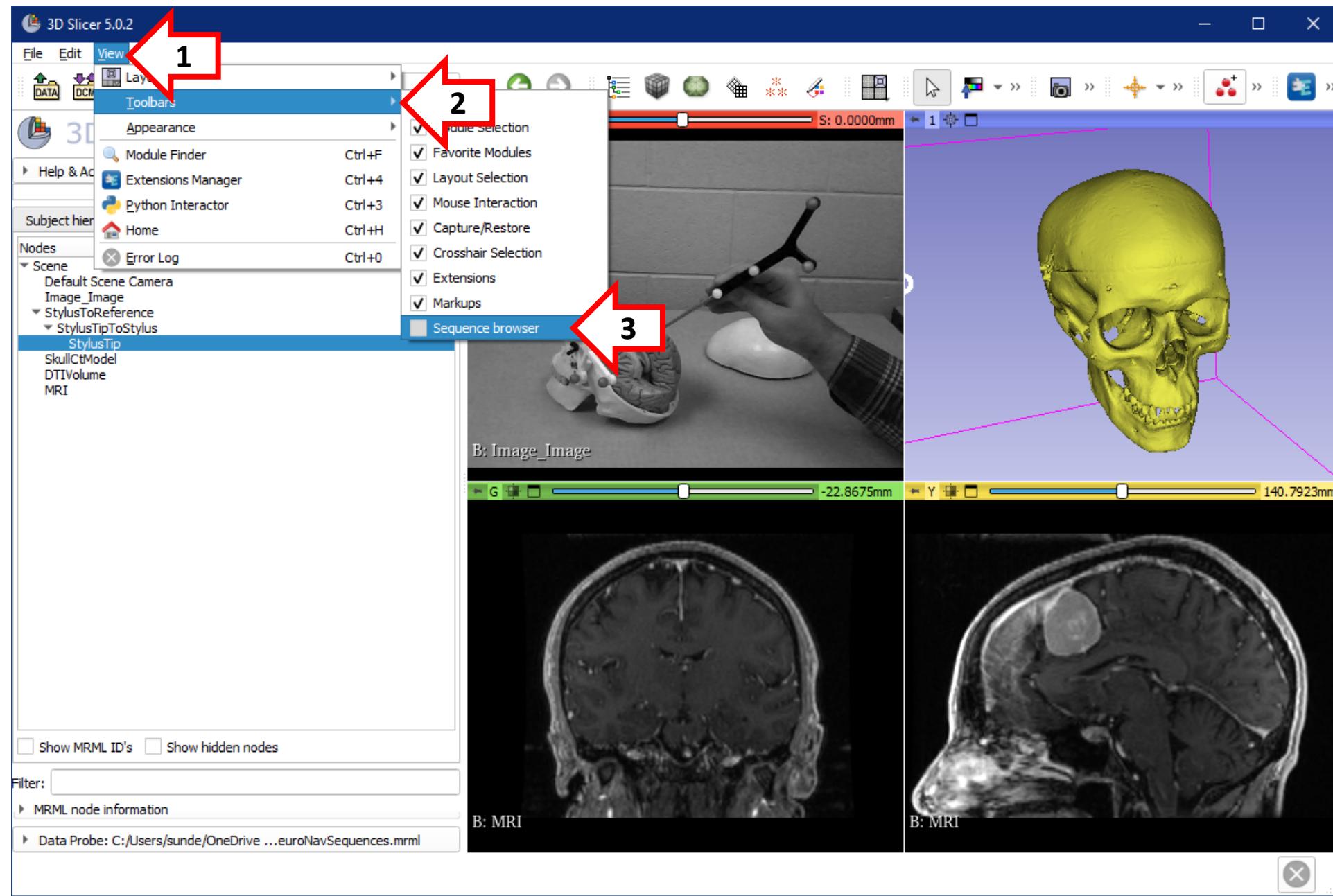


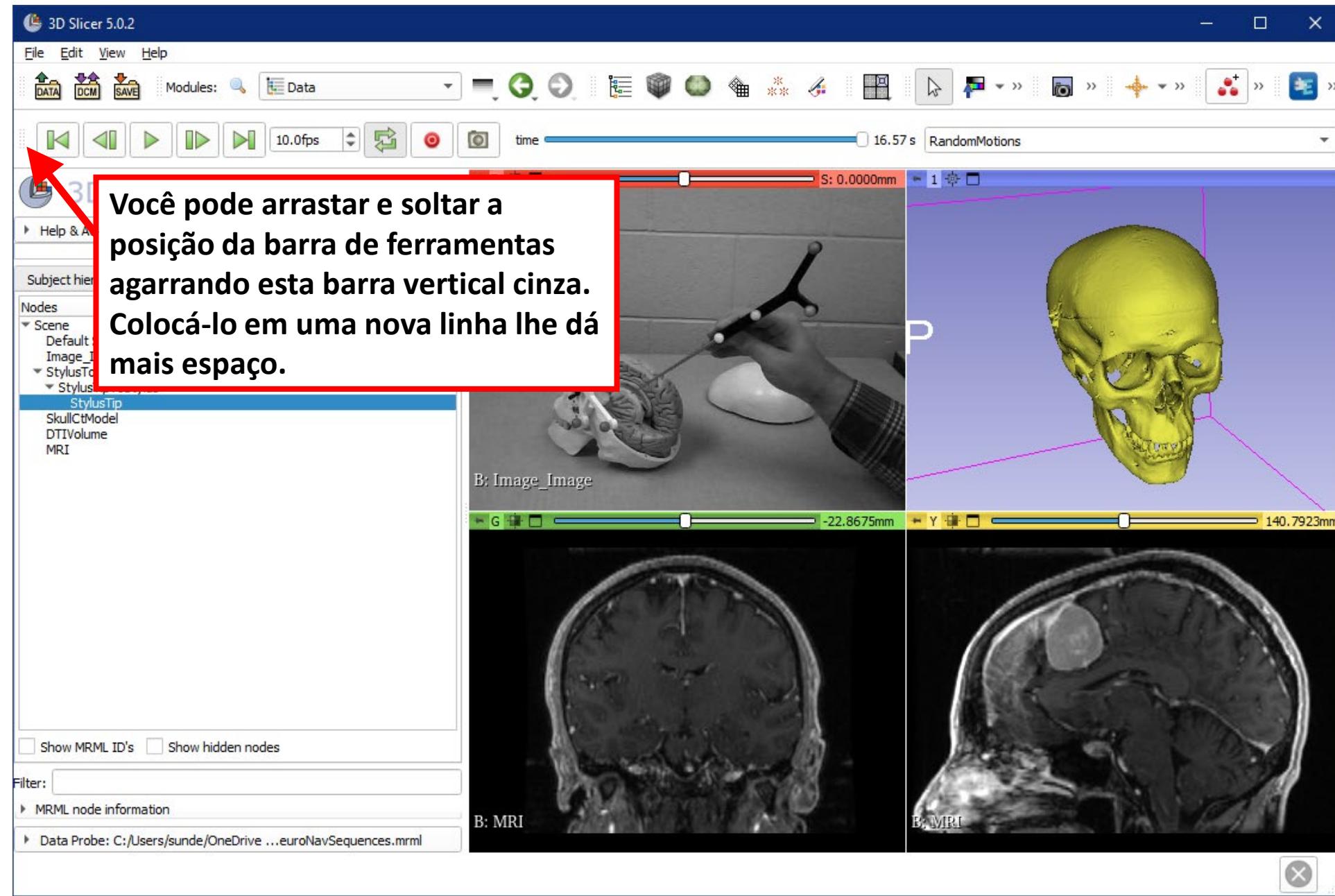


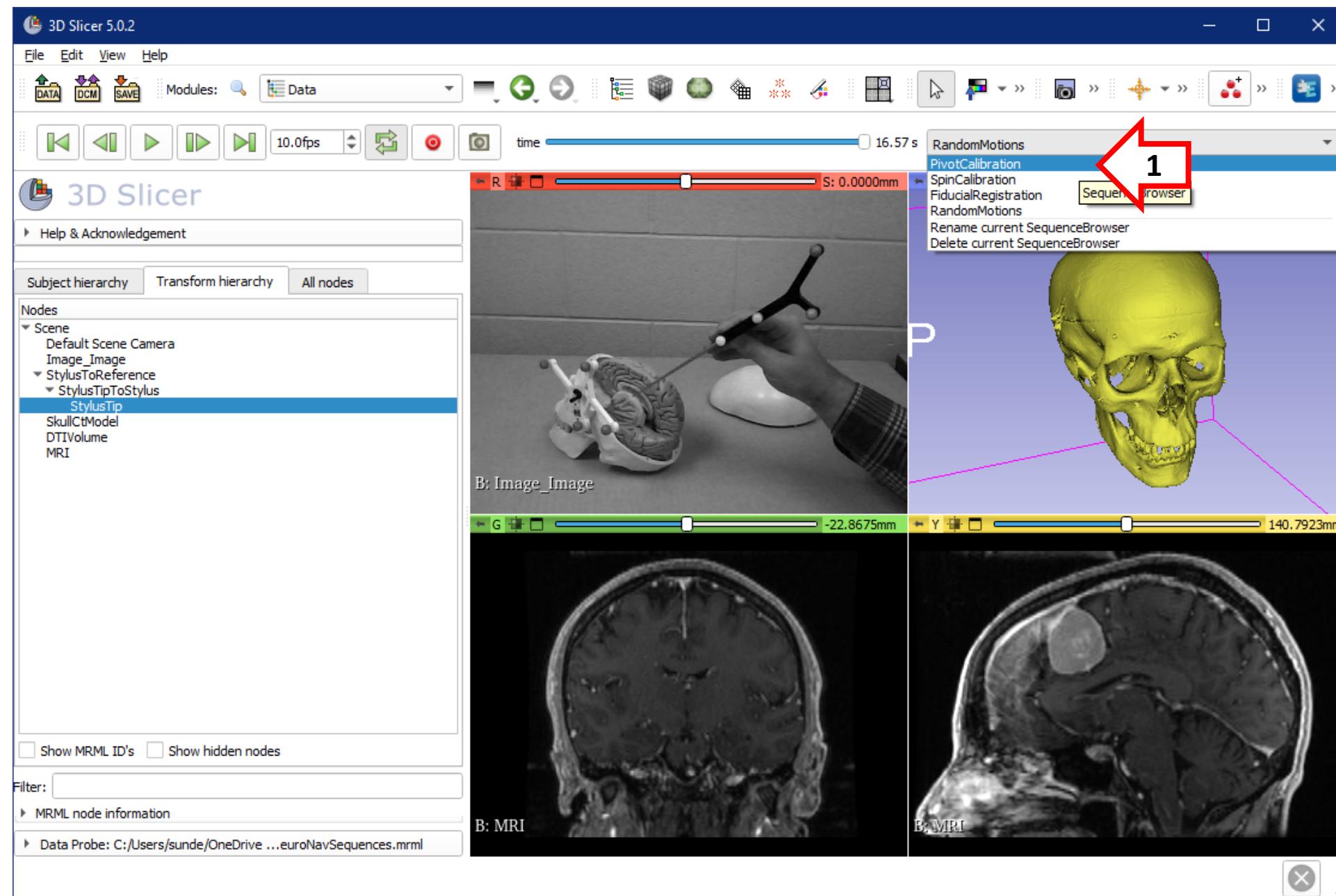


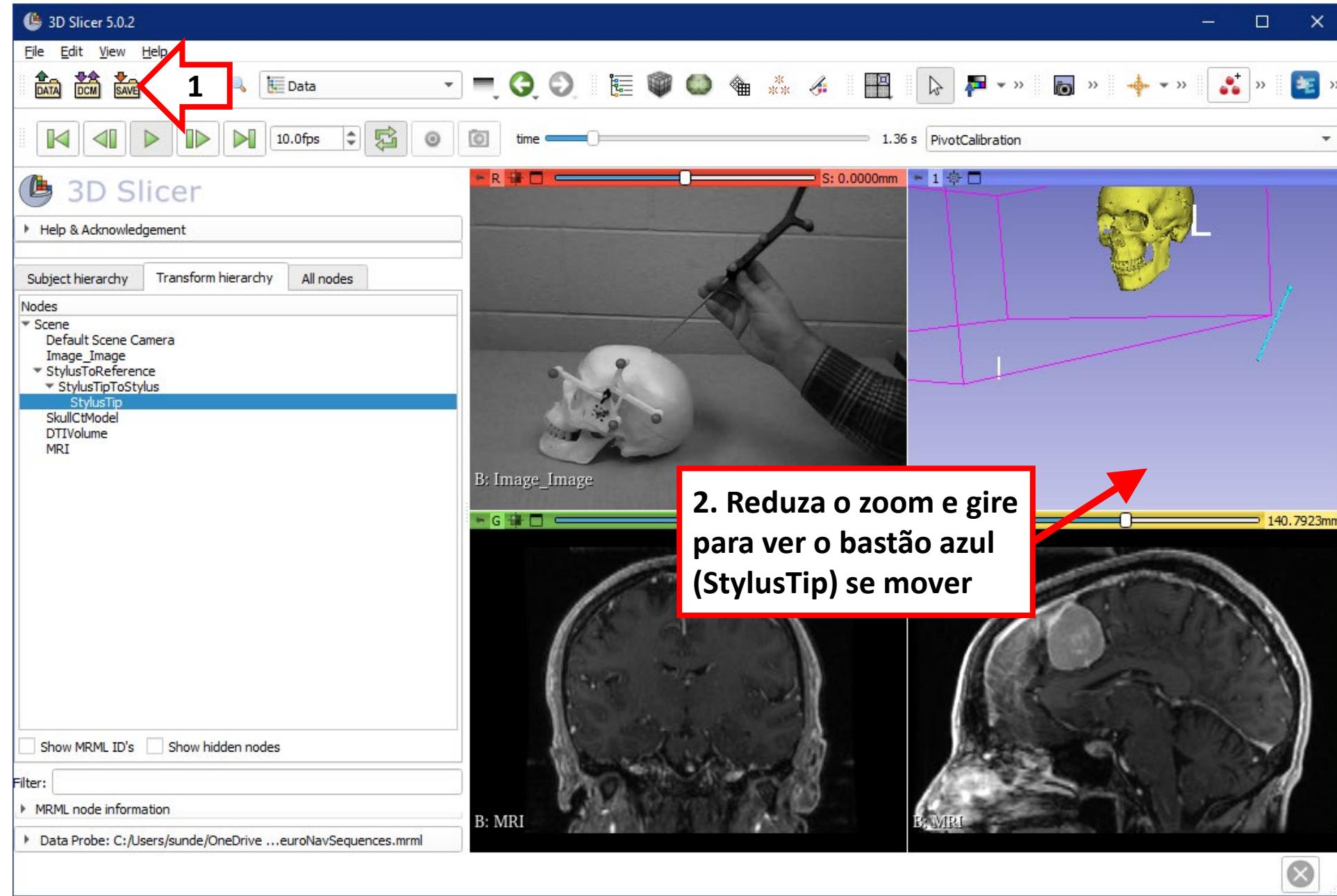


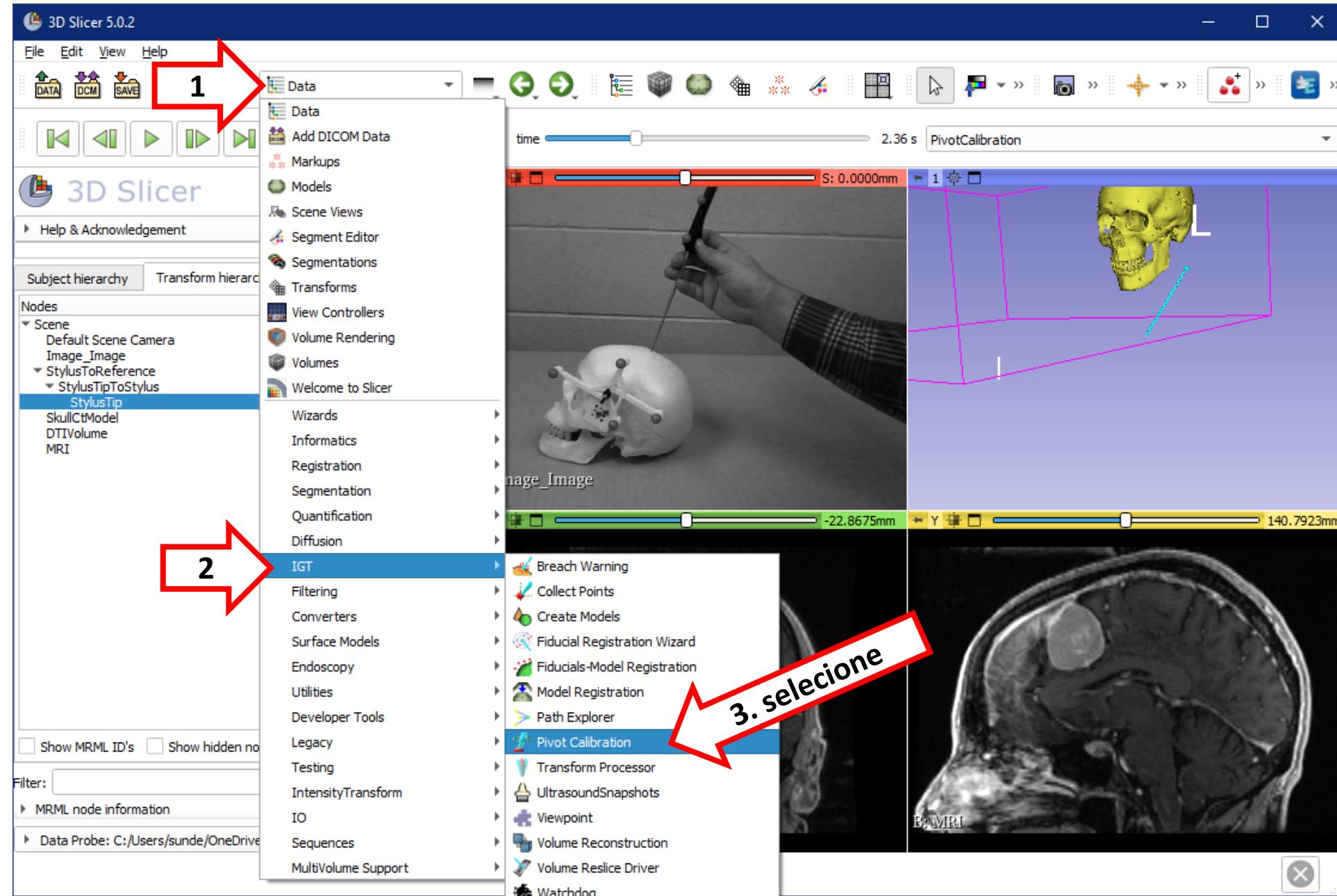


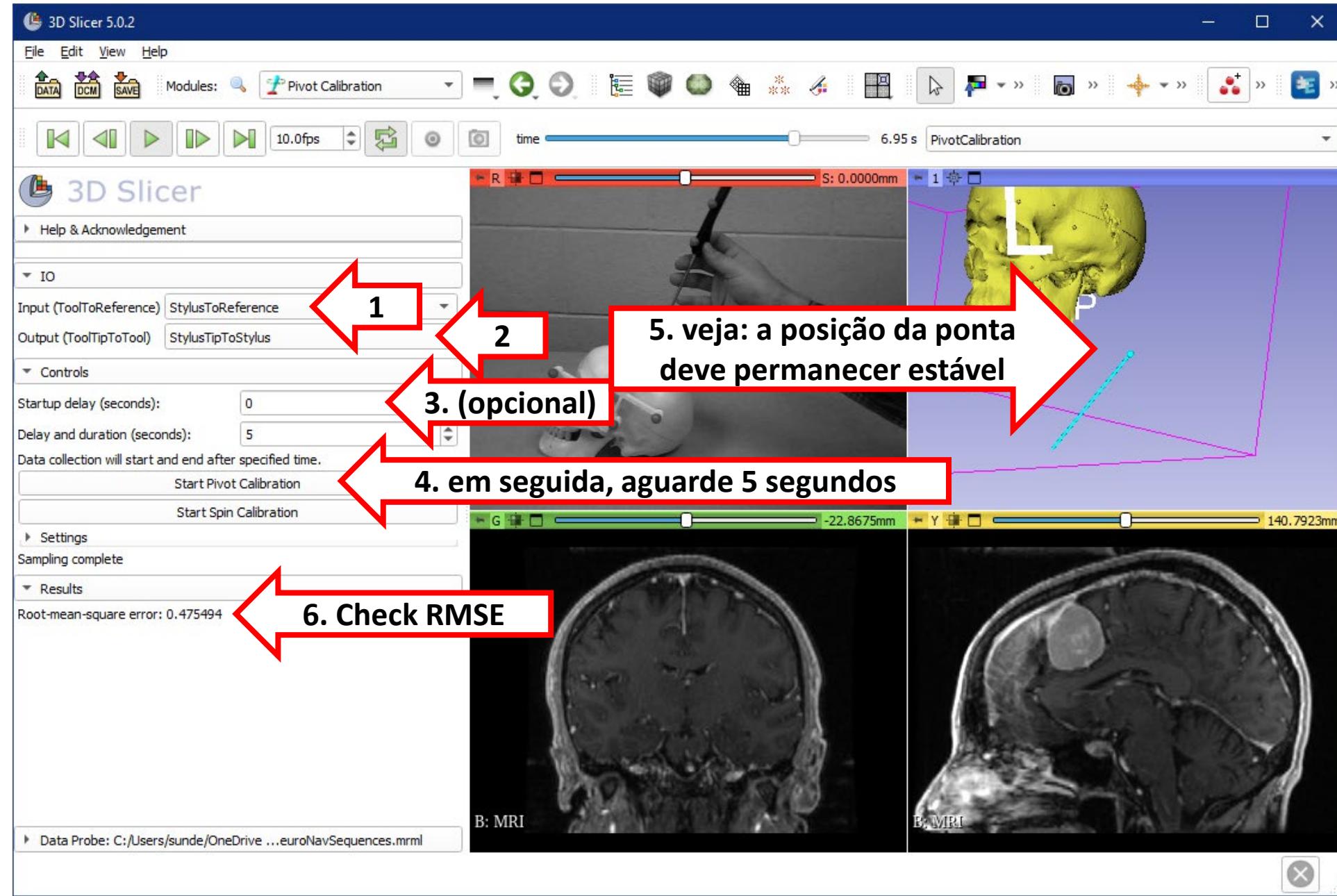


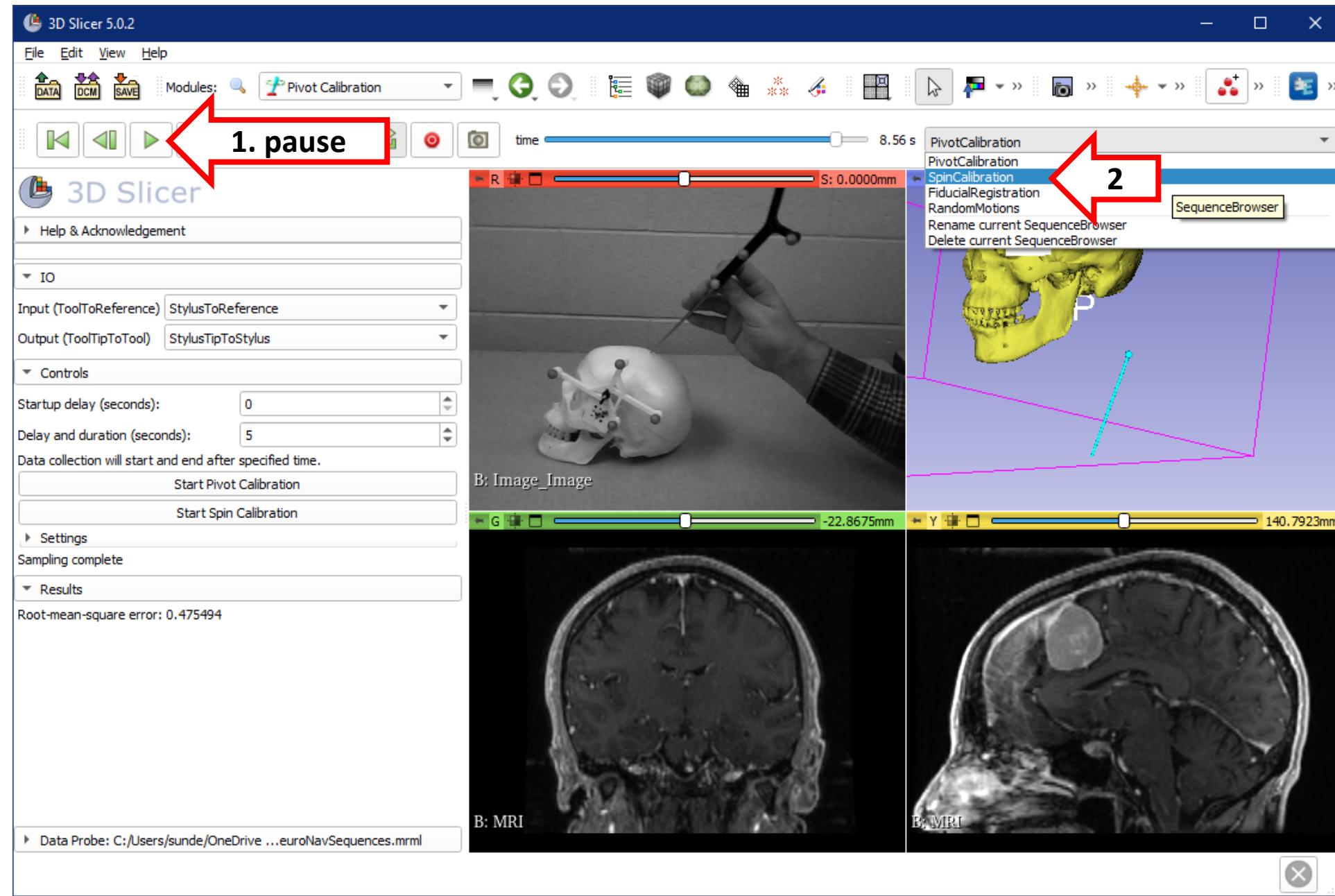


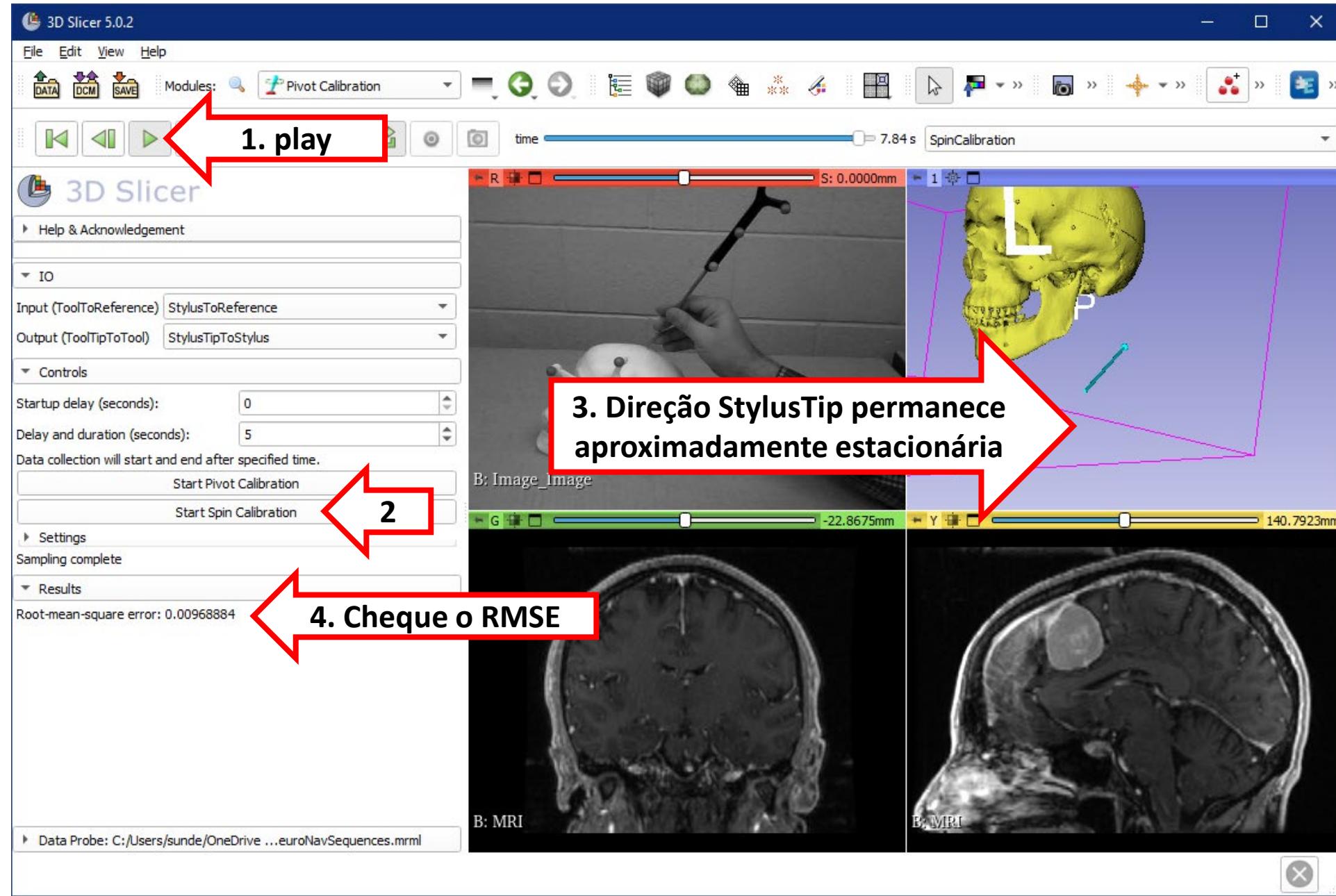


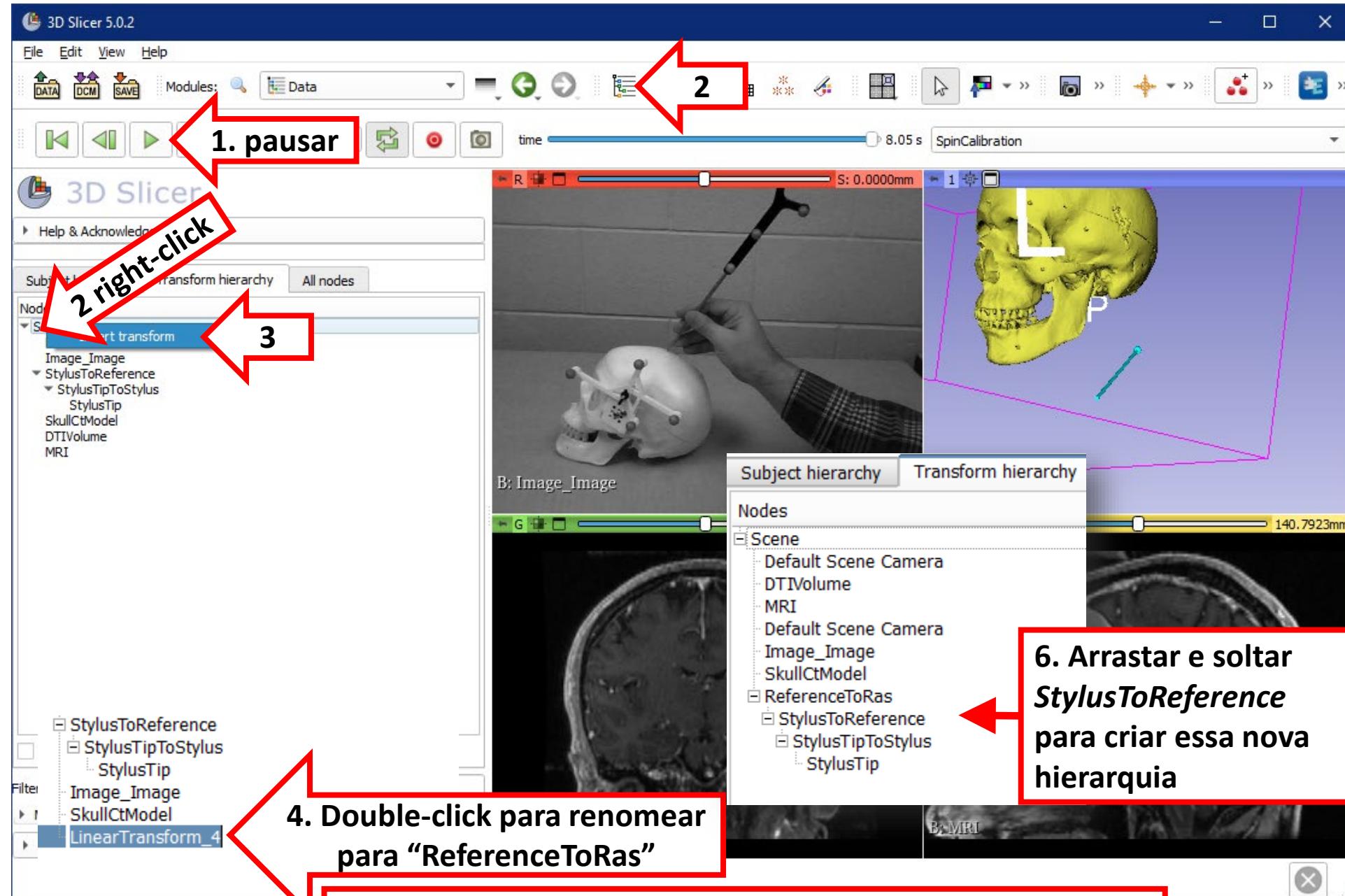


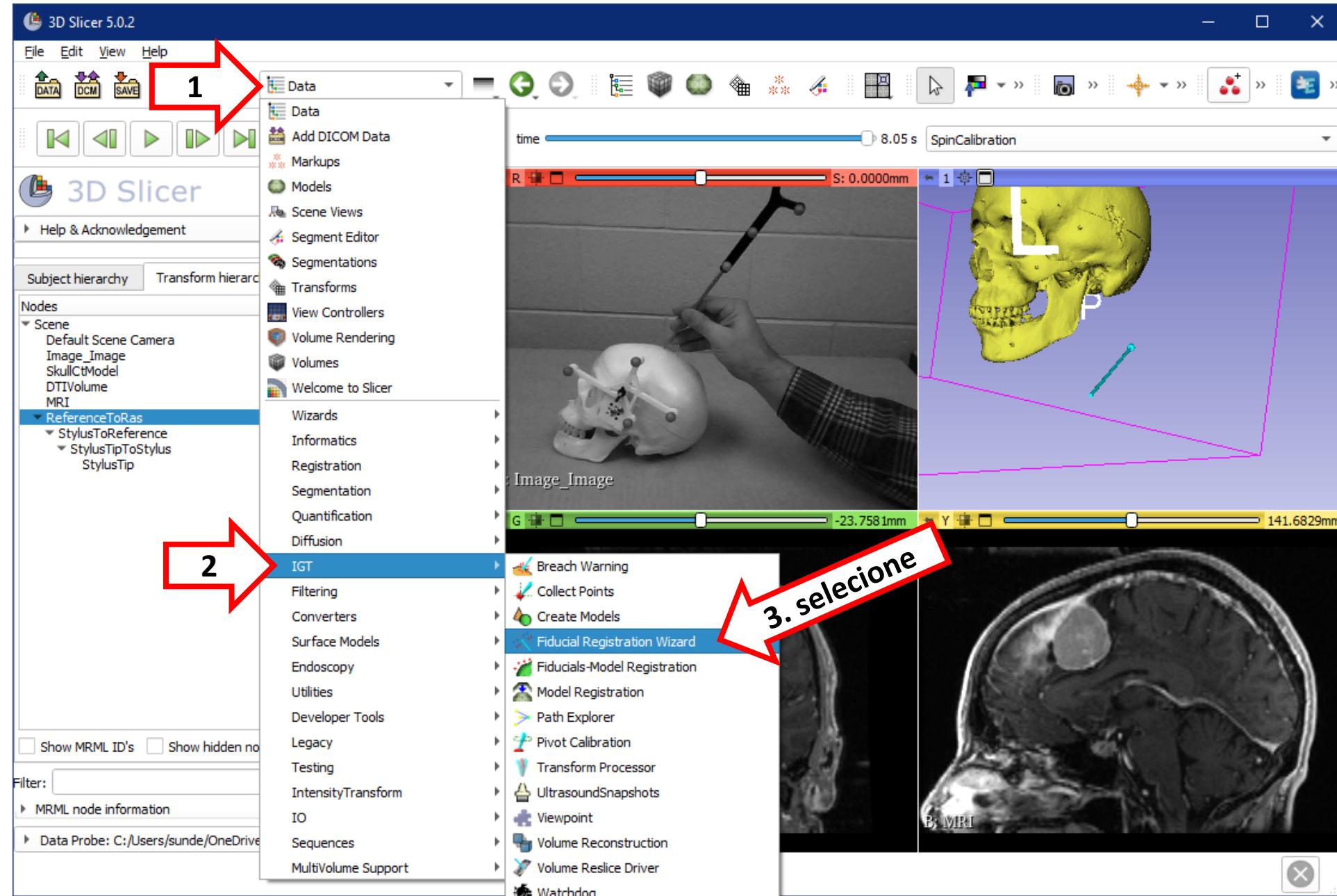


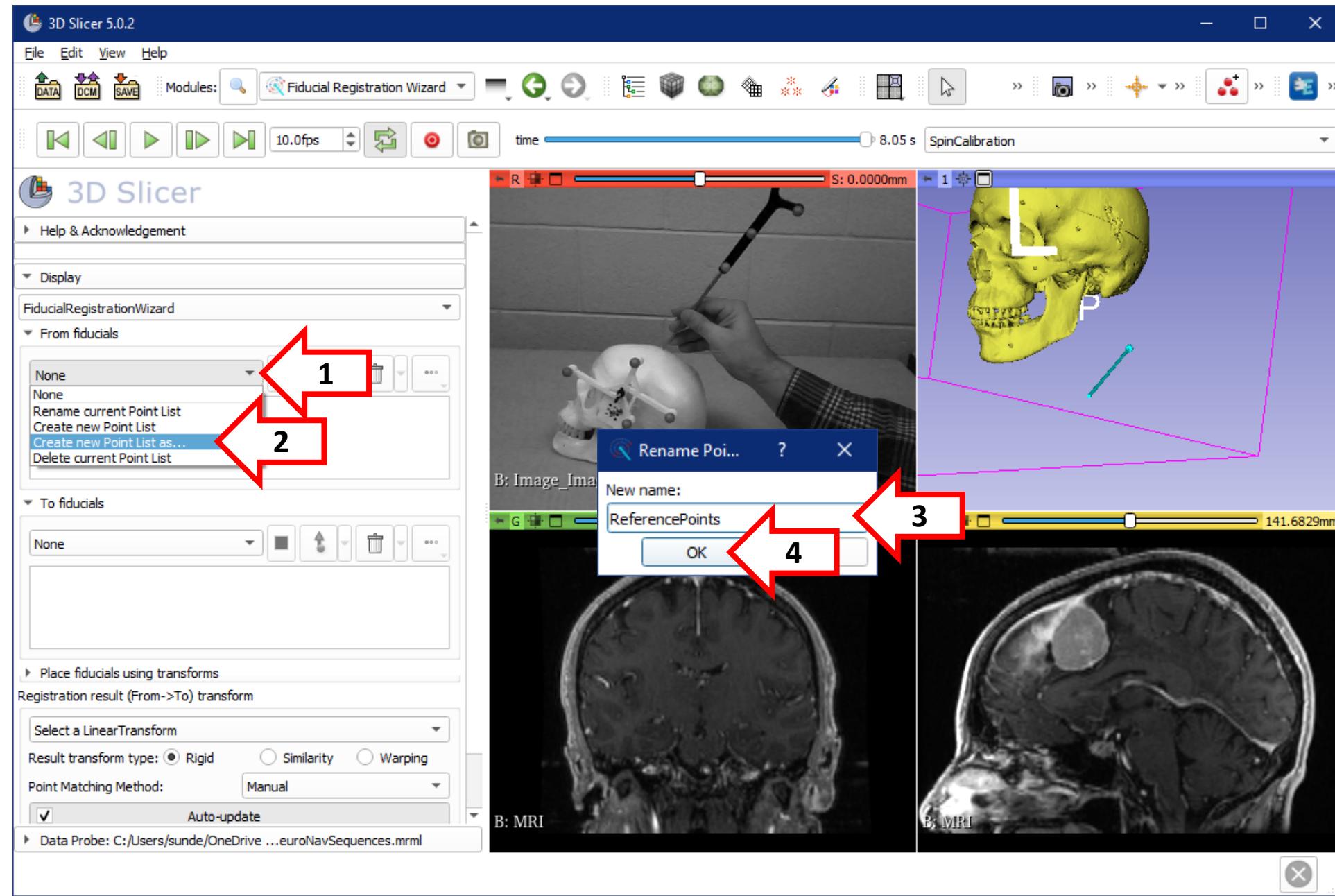


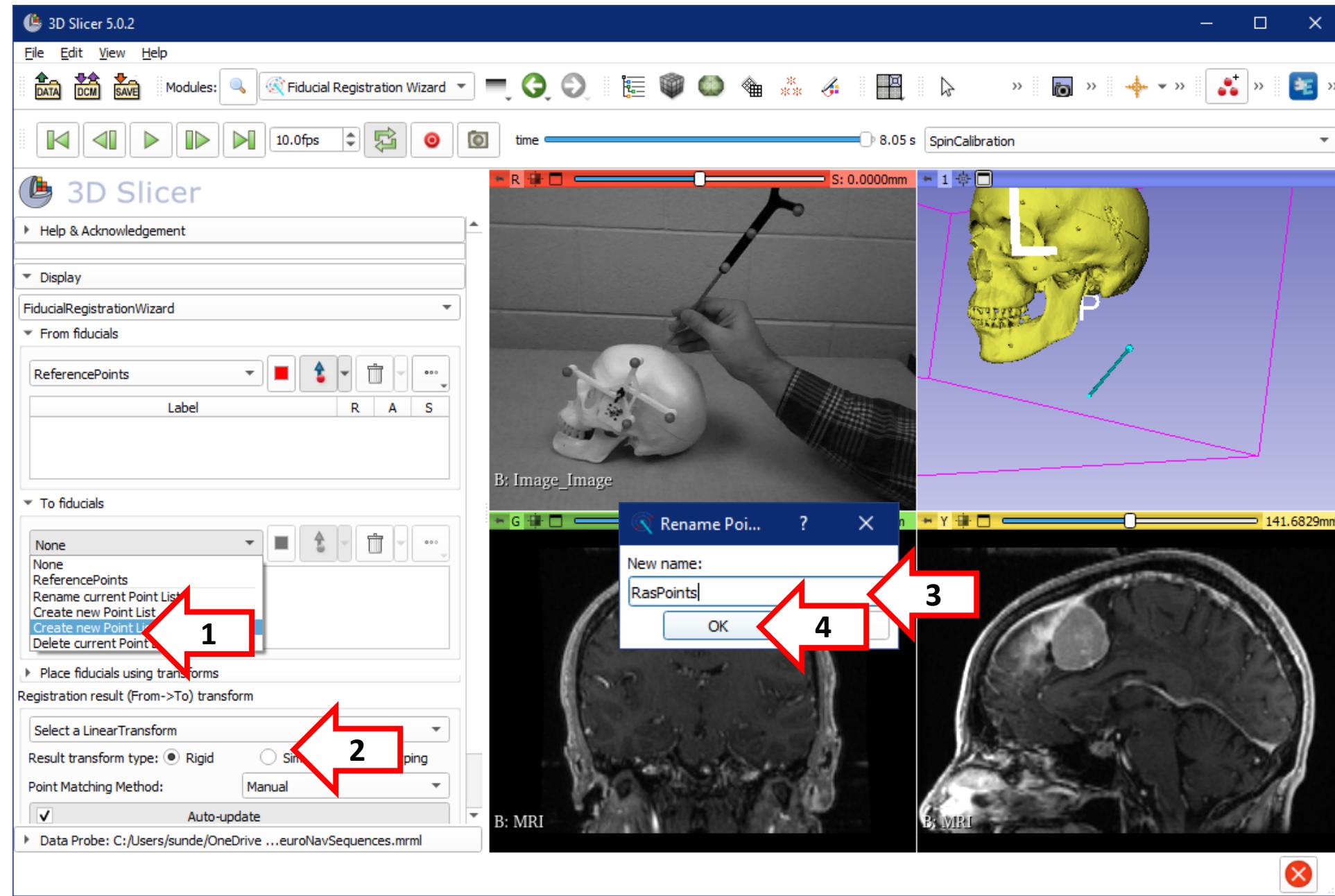


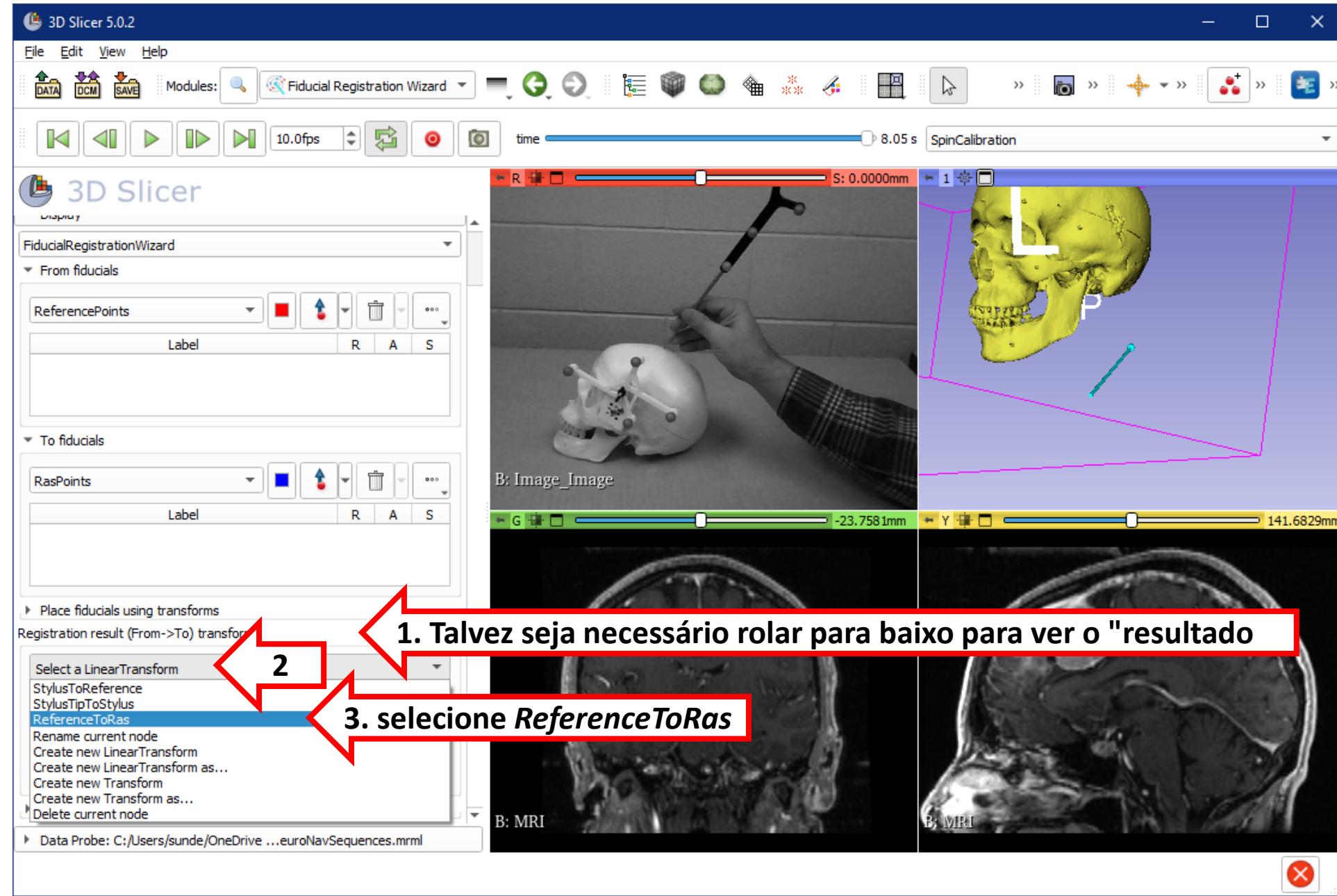


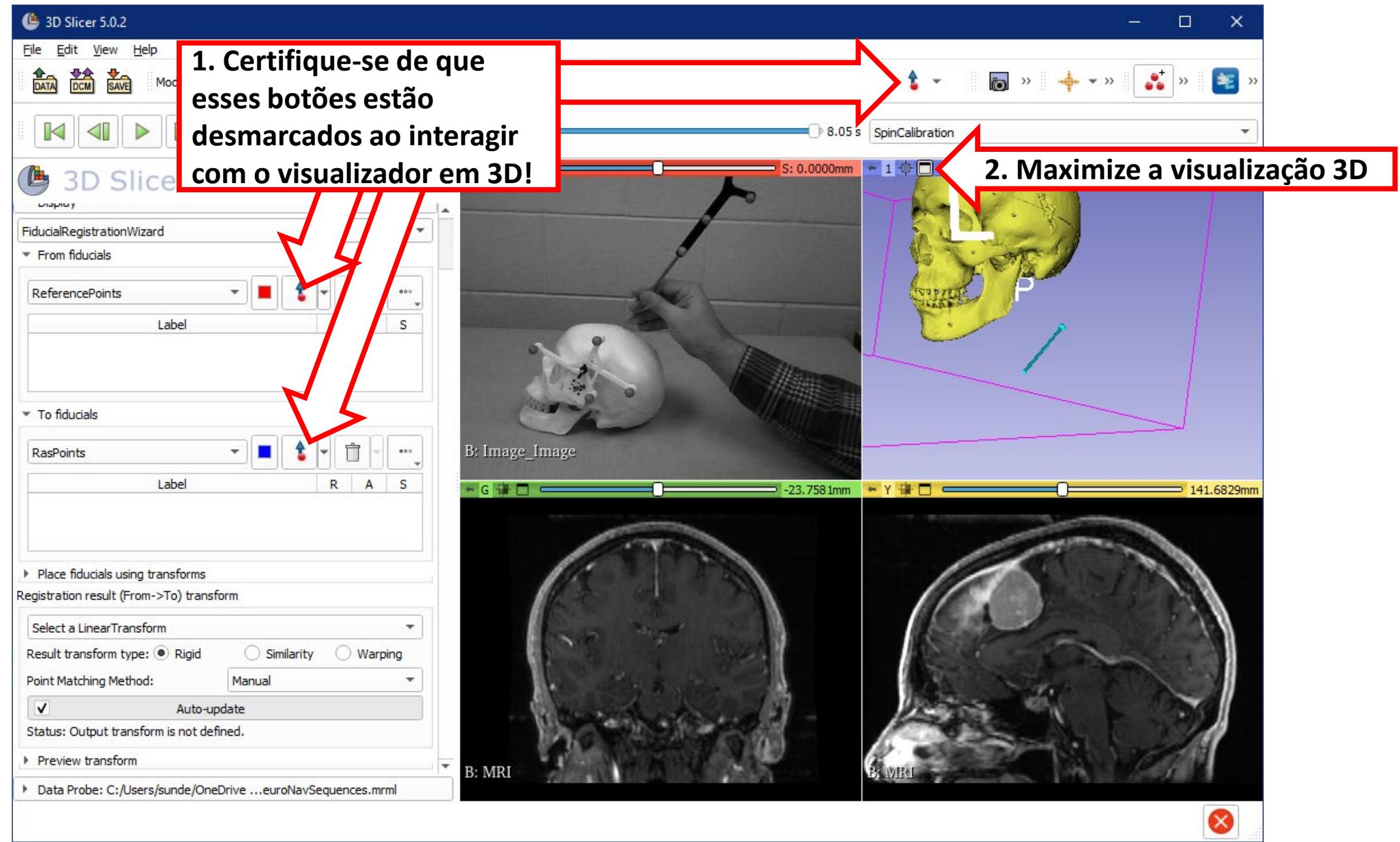


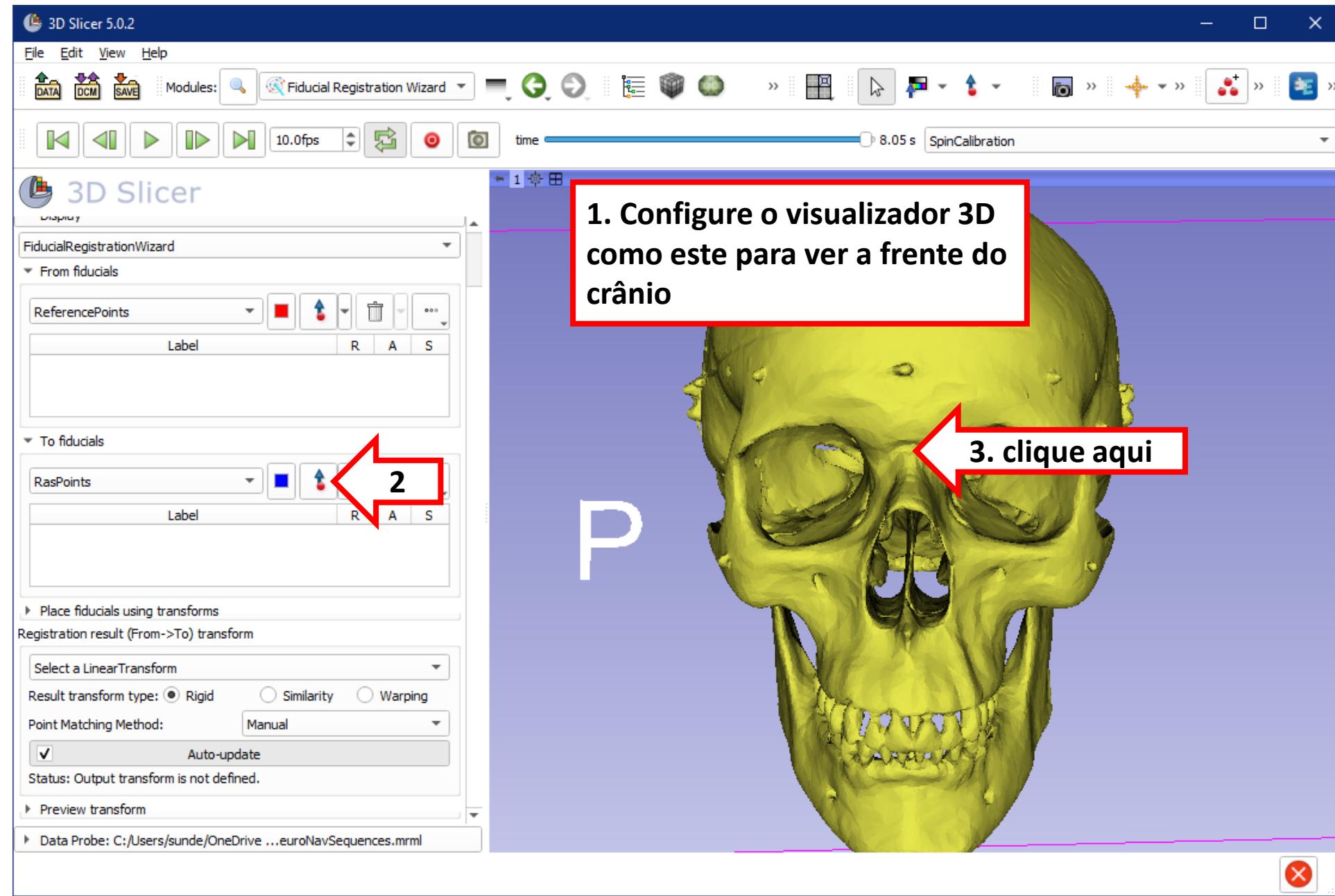


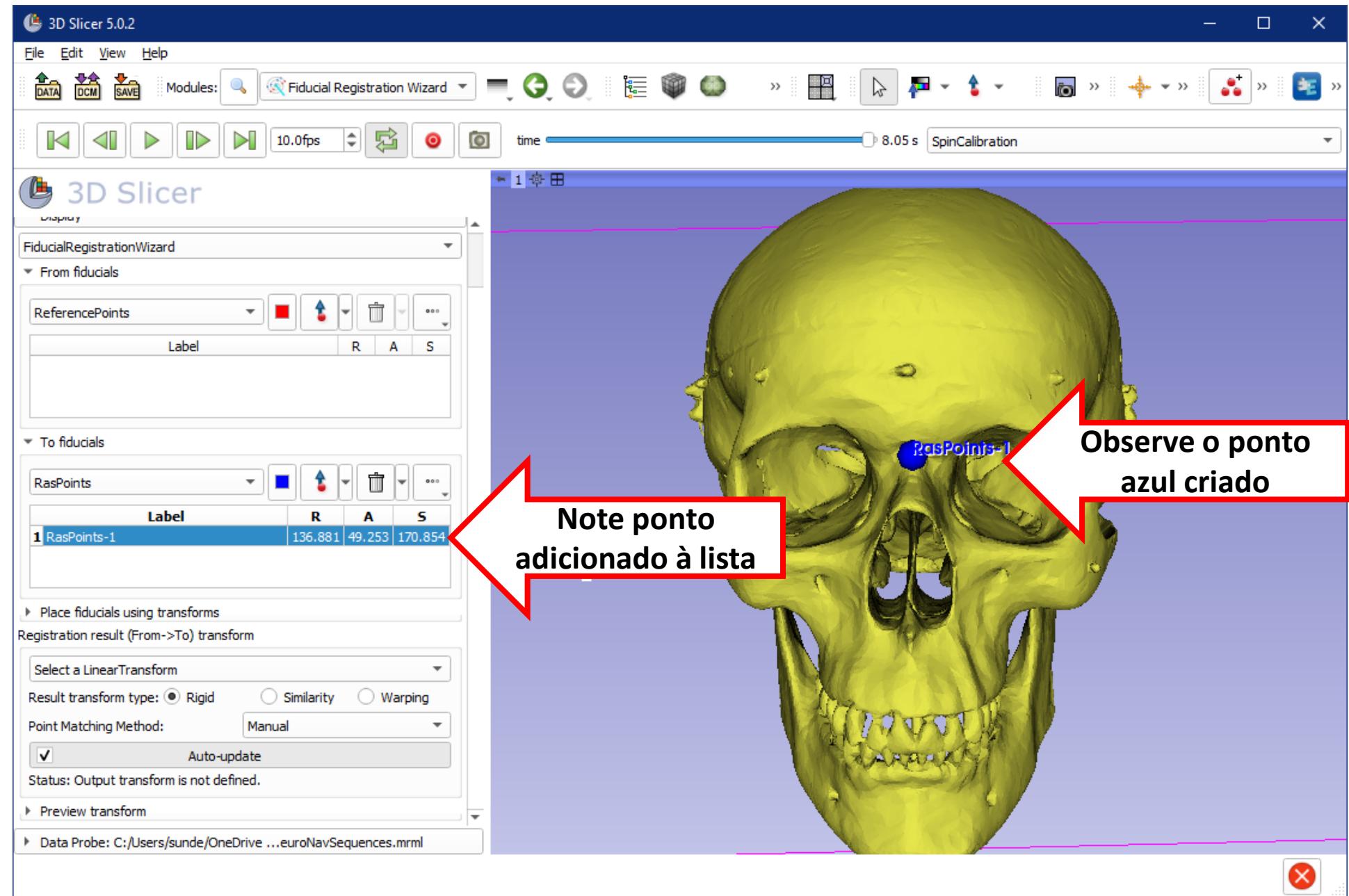


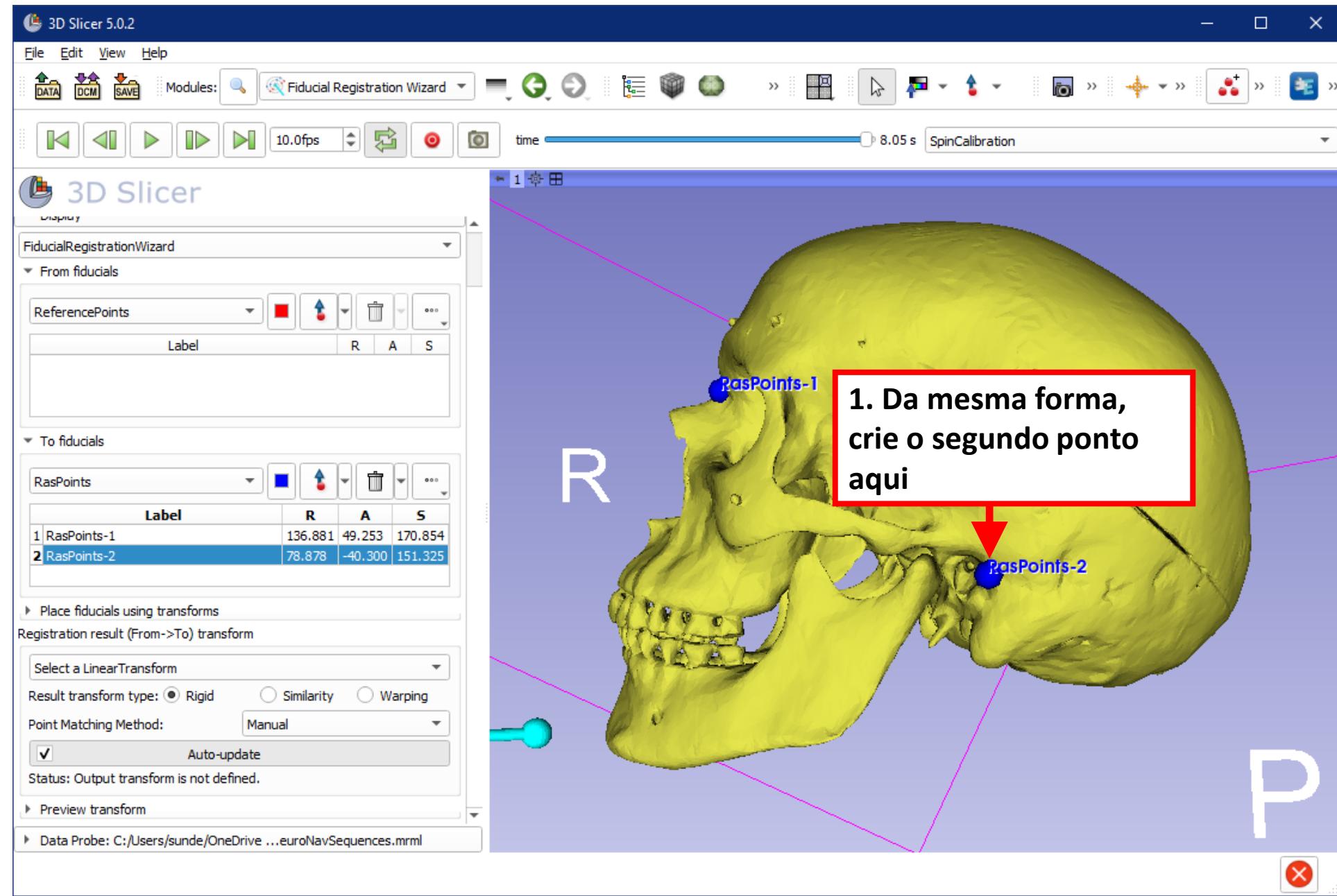


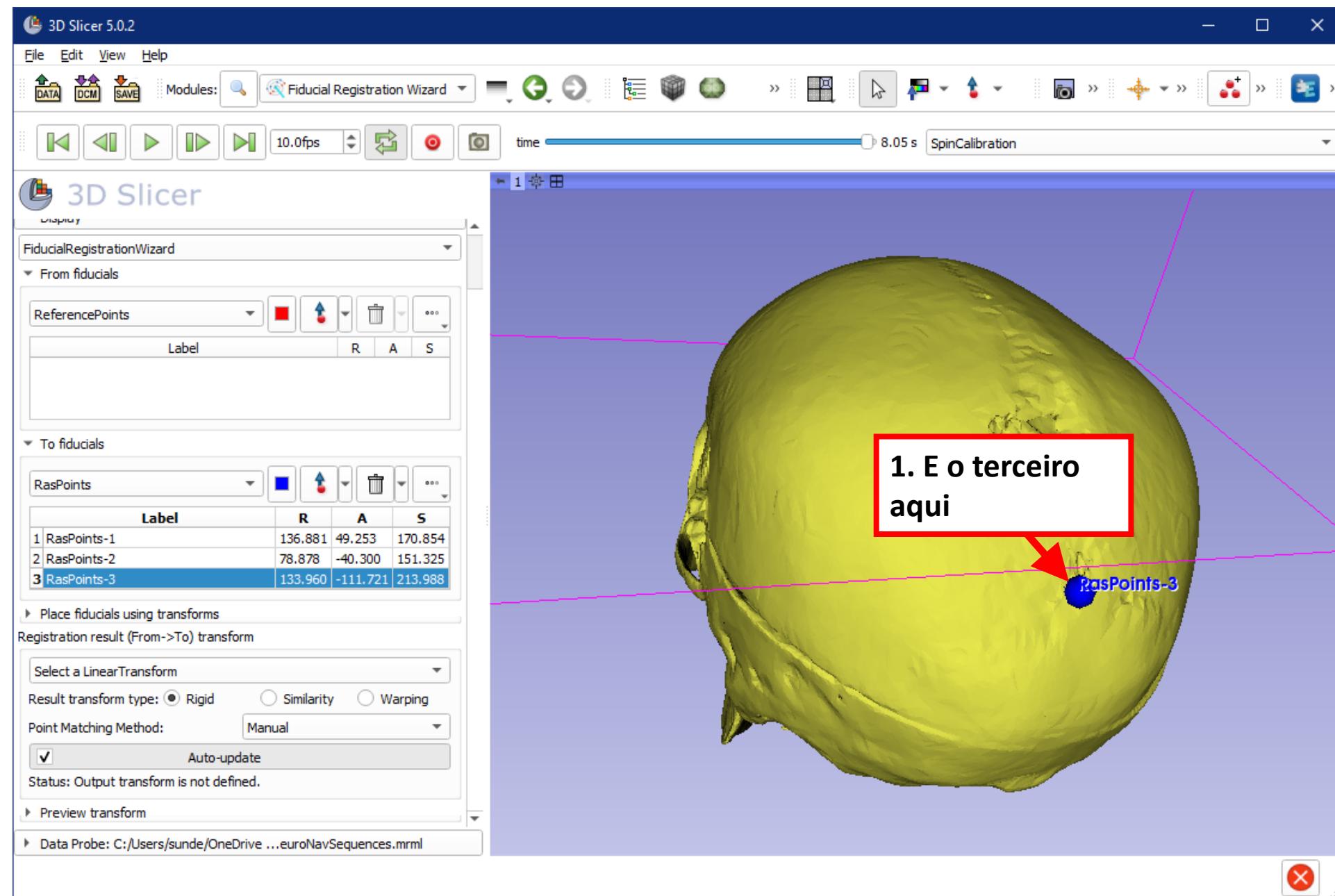


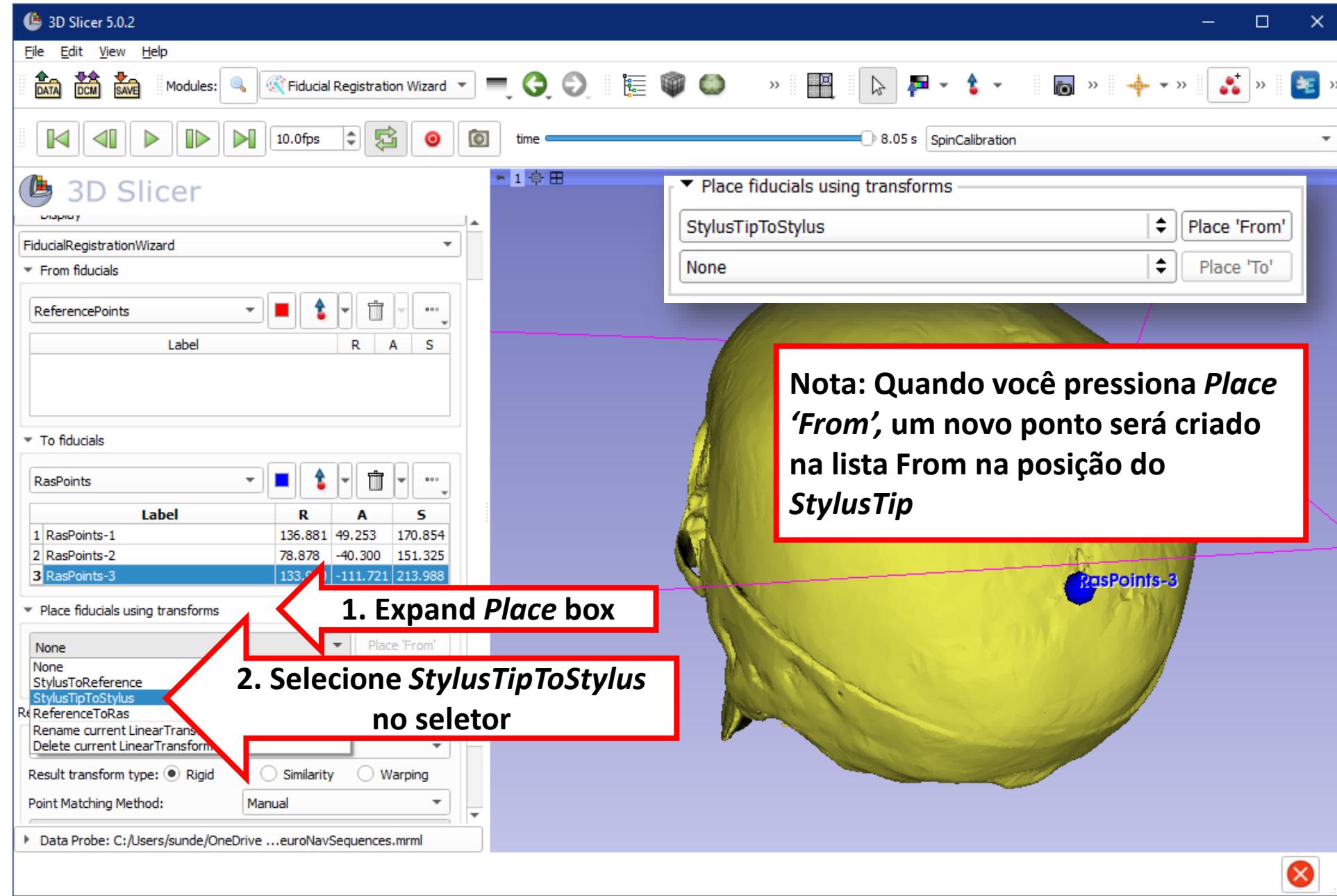


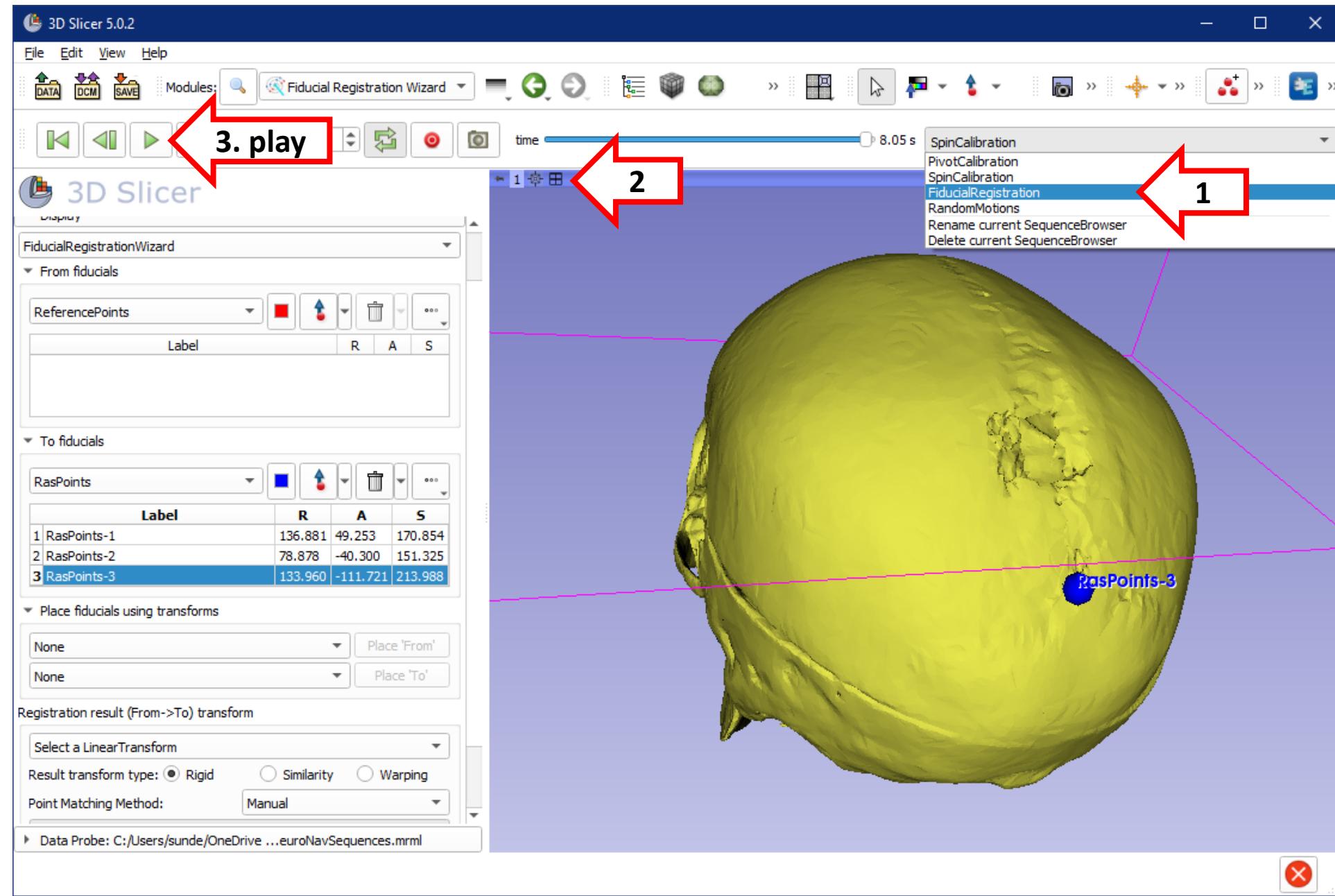


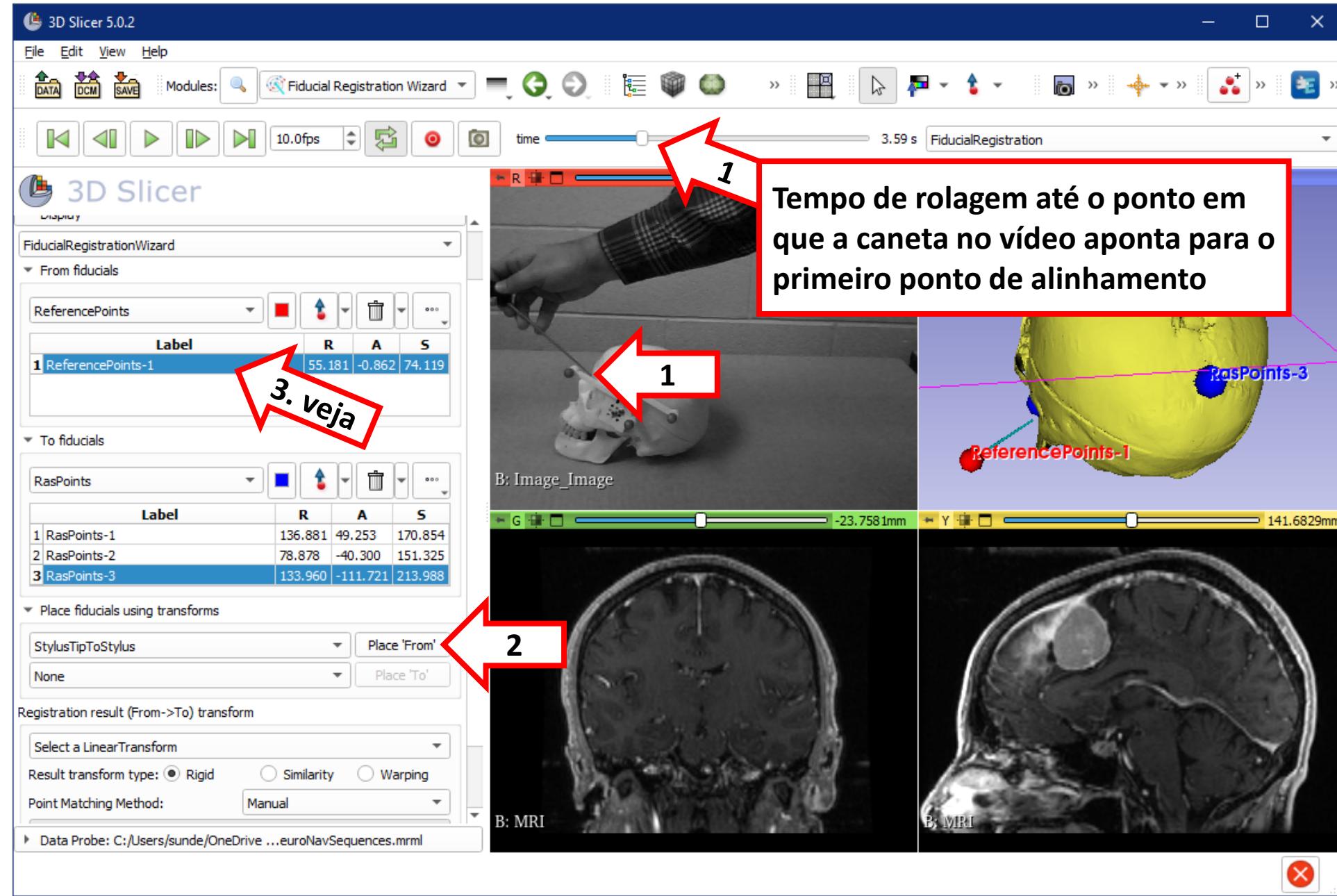


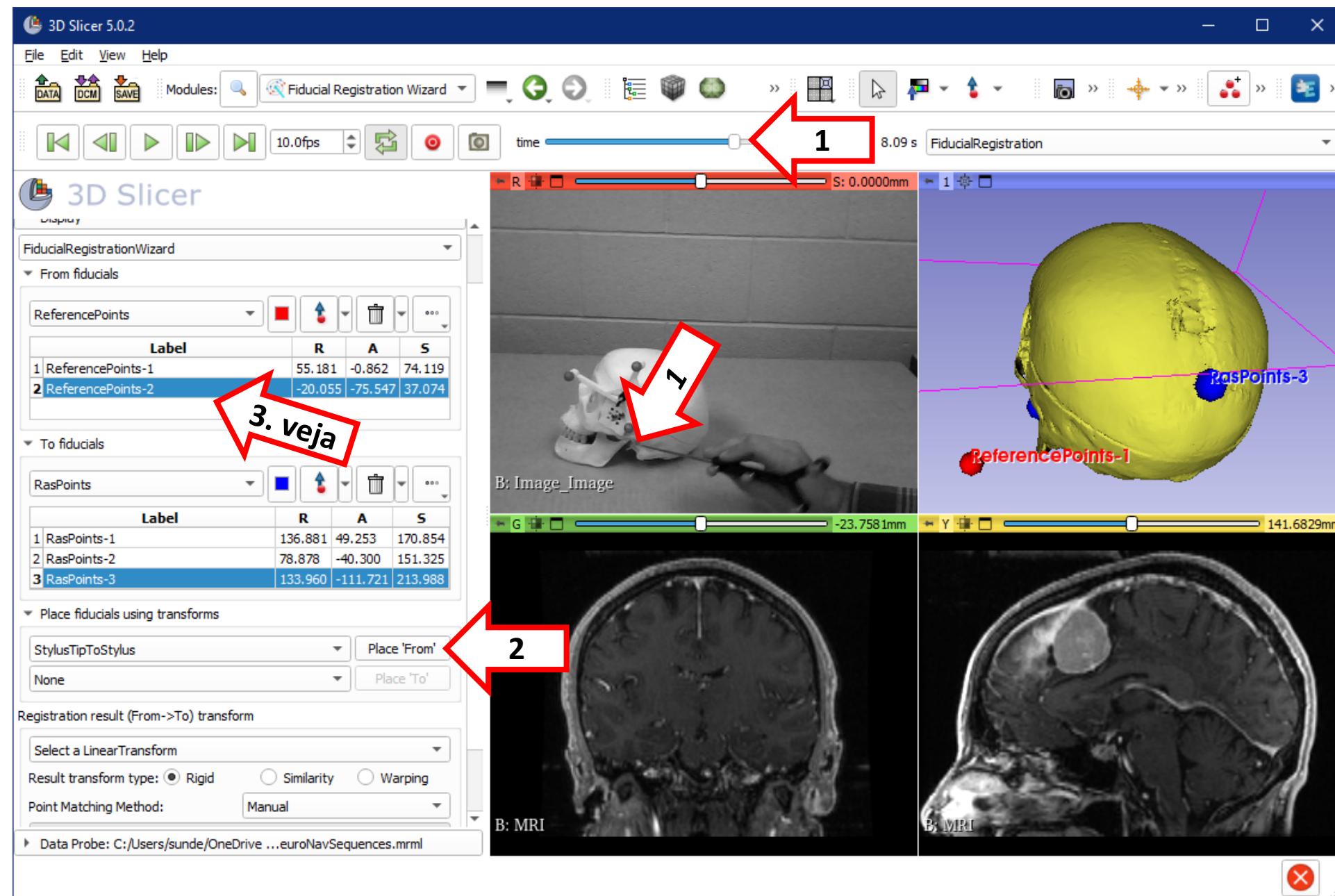


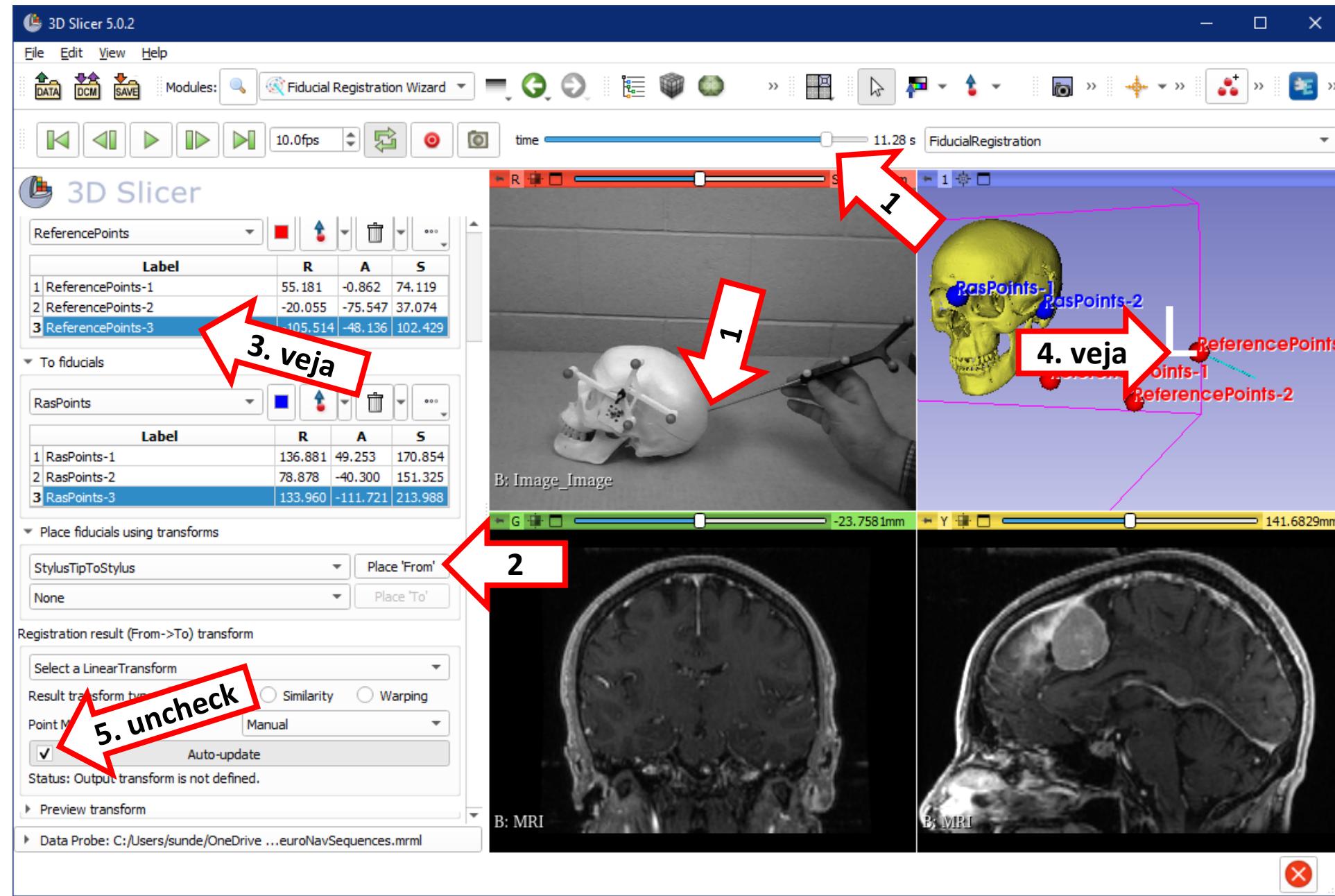






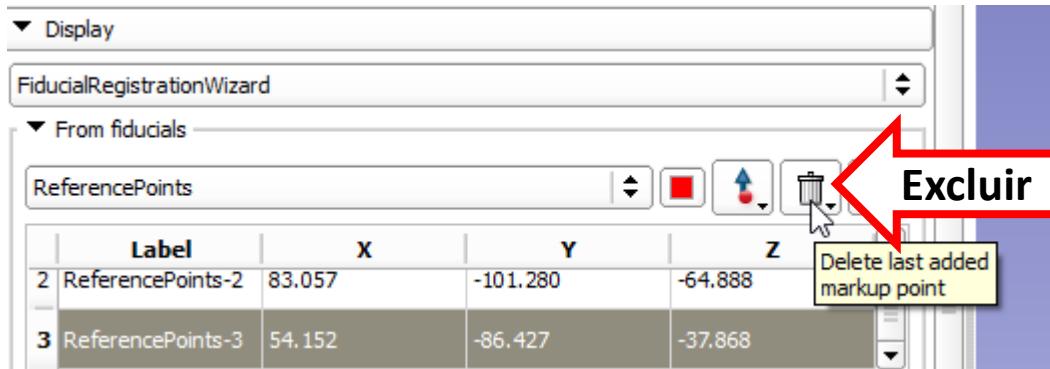




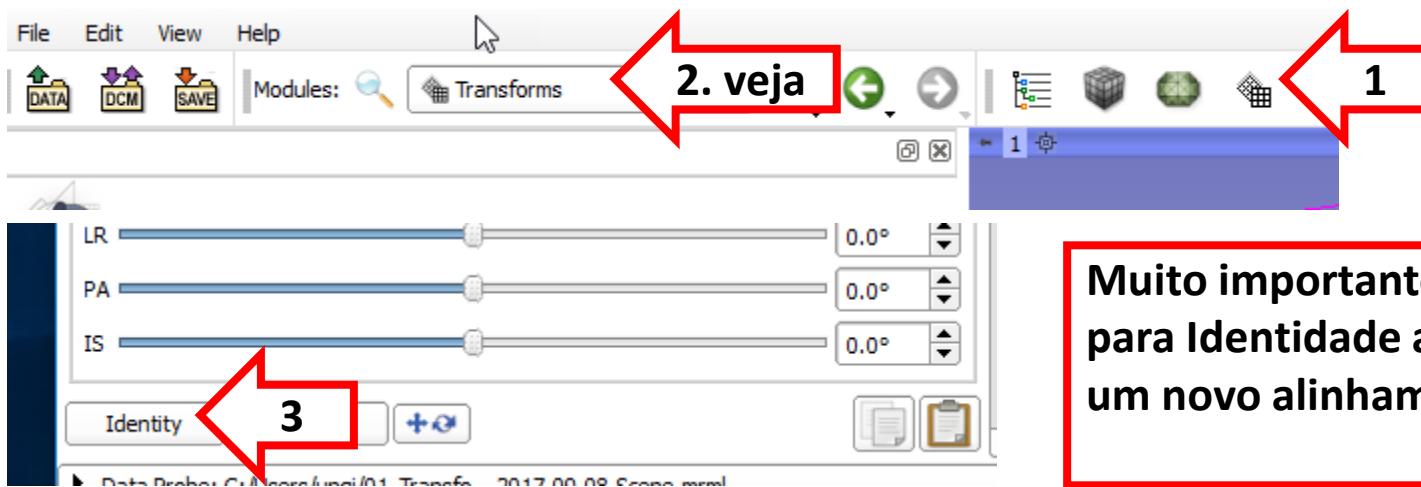


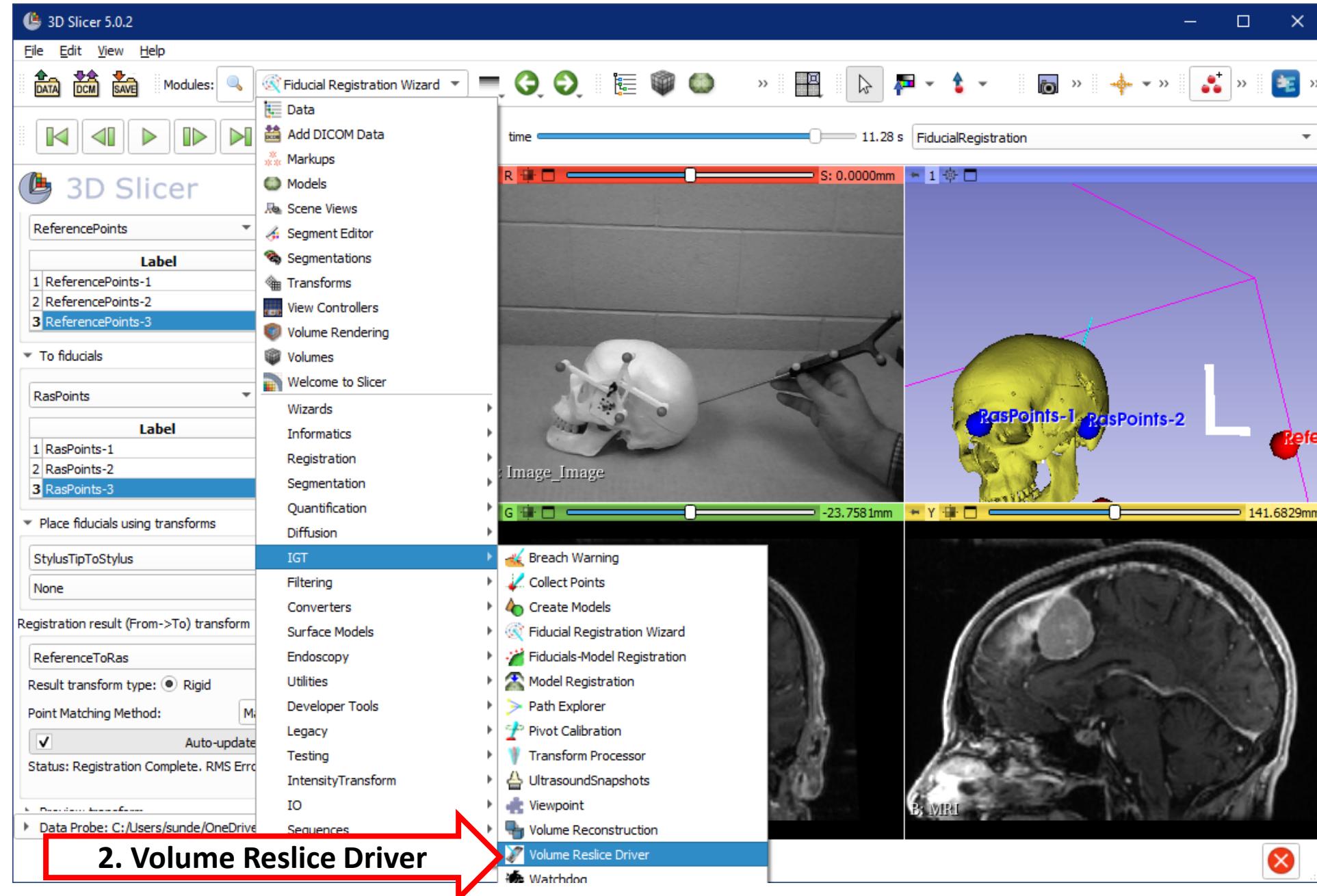
Troubleshooting

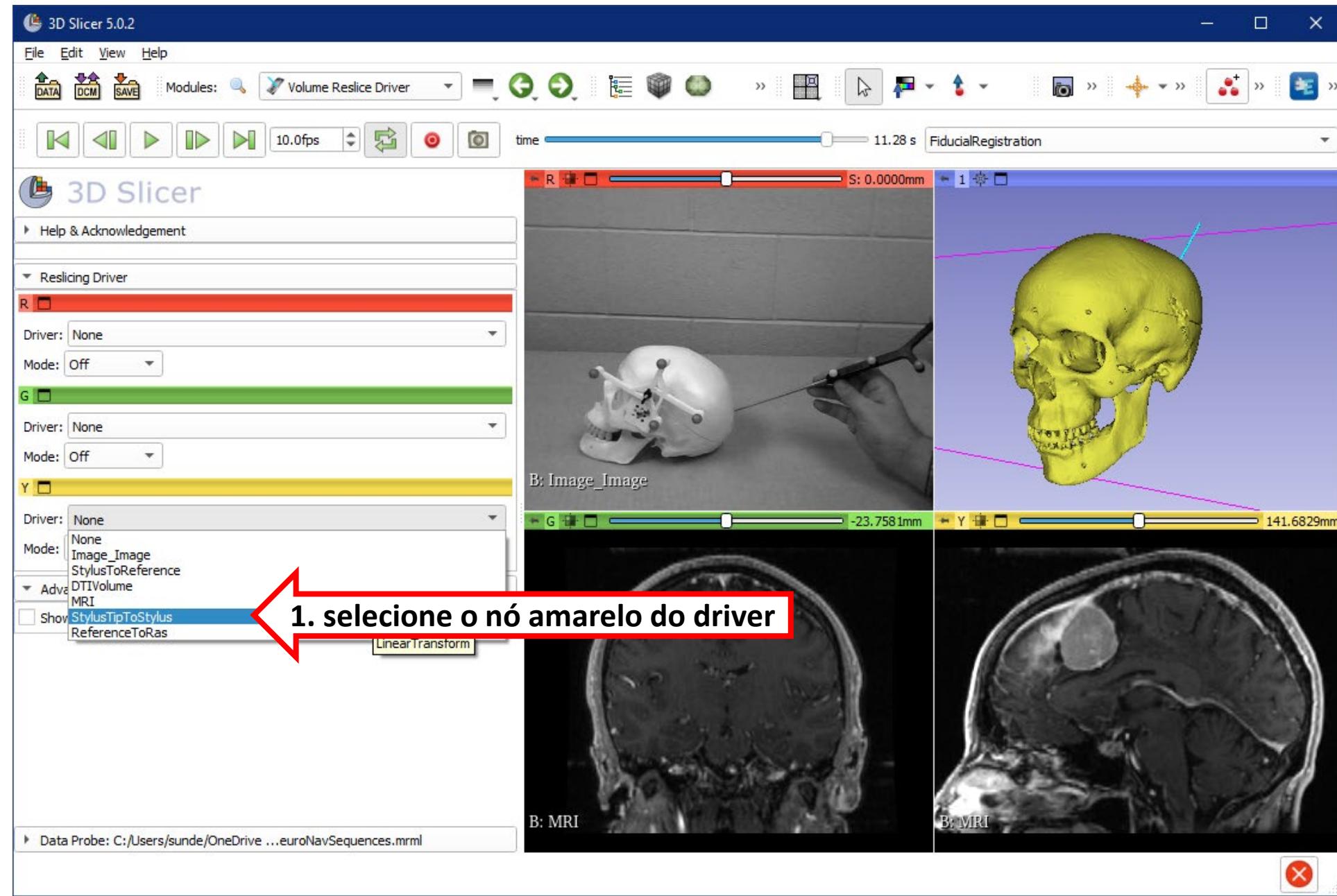
How to redo one point (before registration completed)

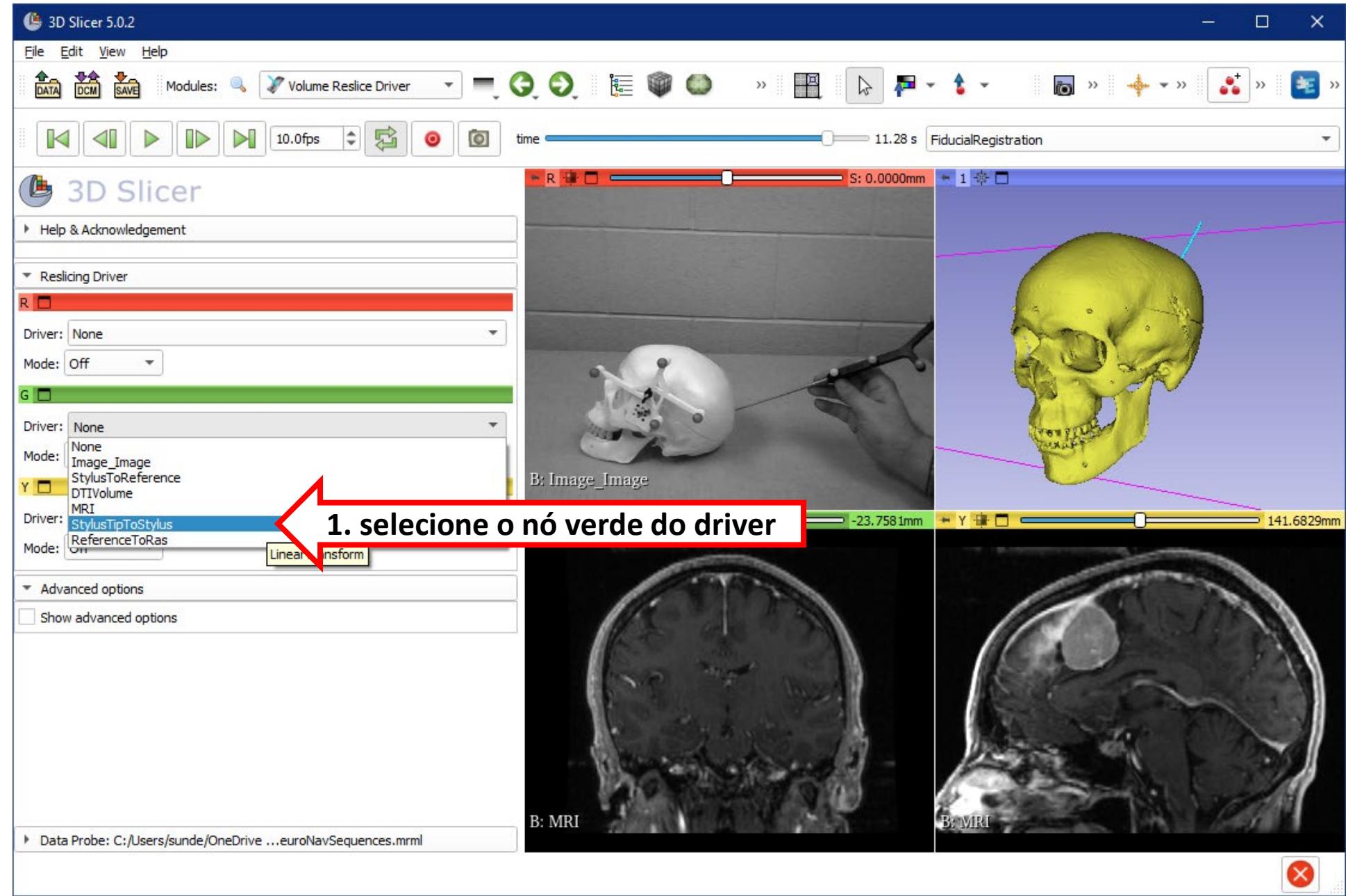


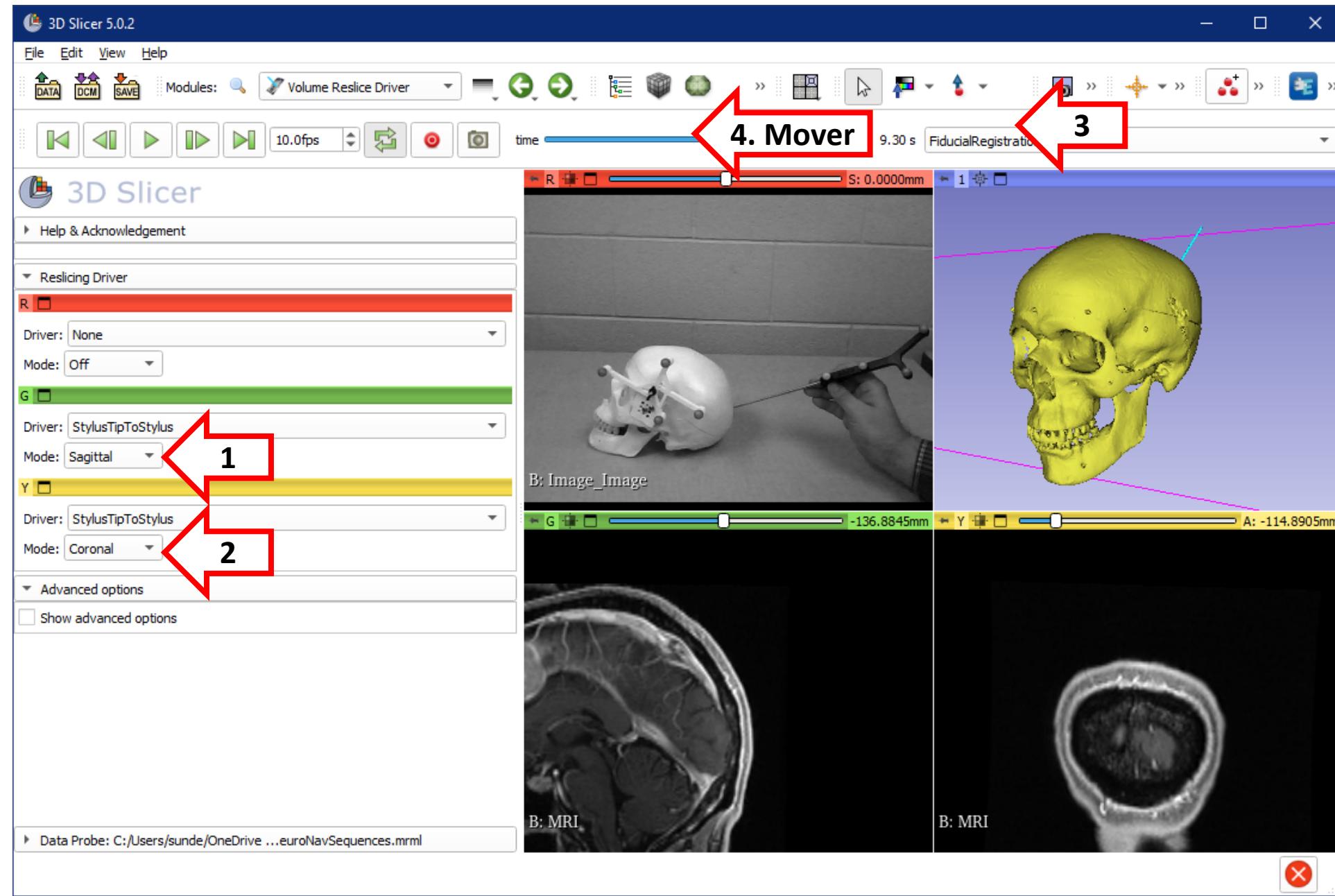
How to reset registration (after a completed registration)

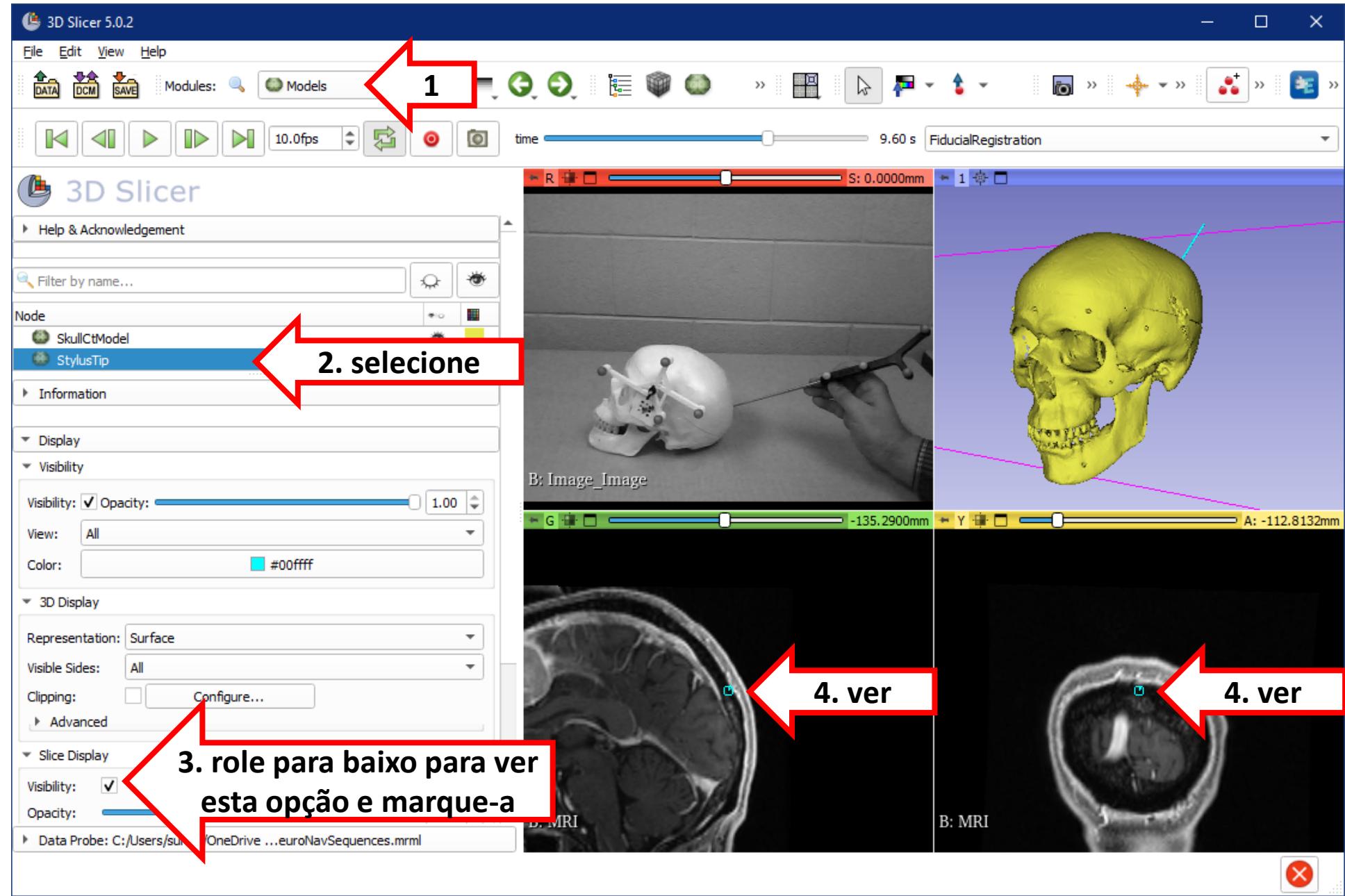


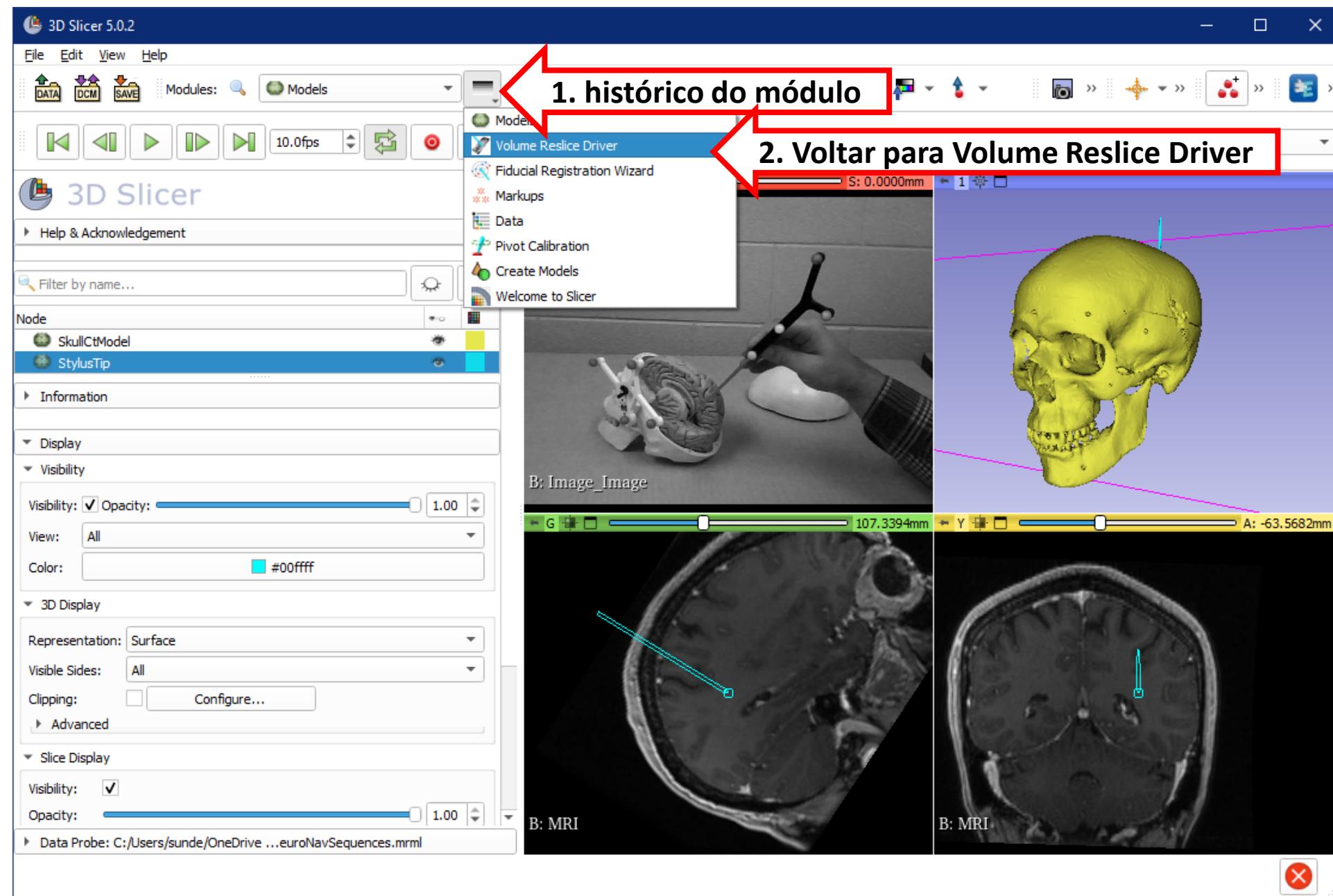


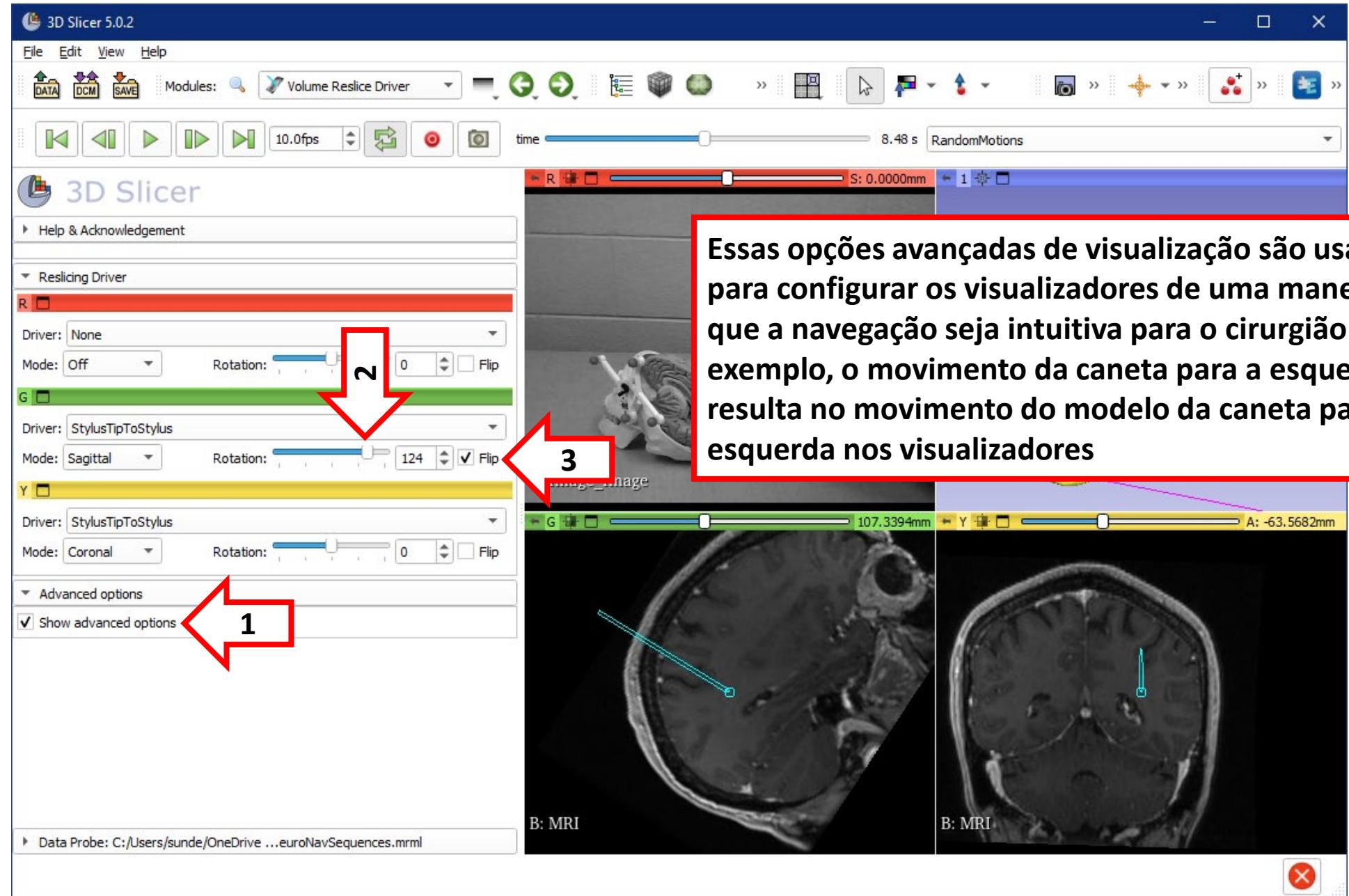


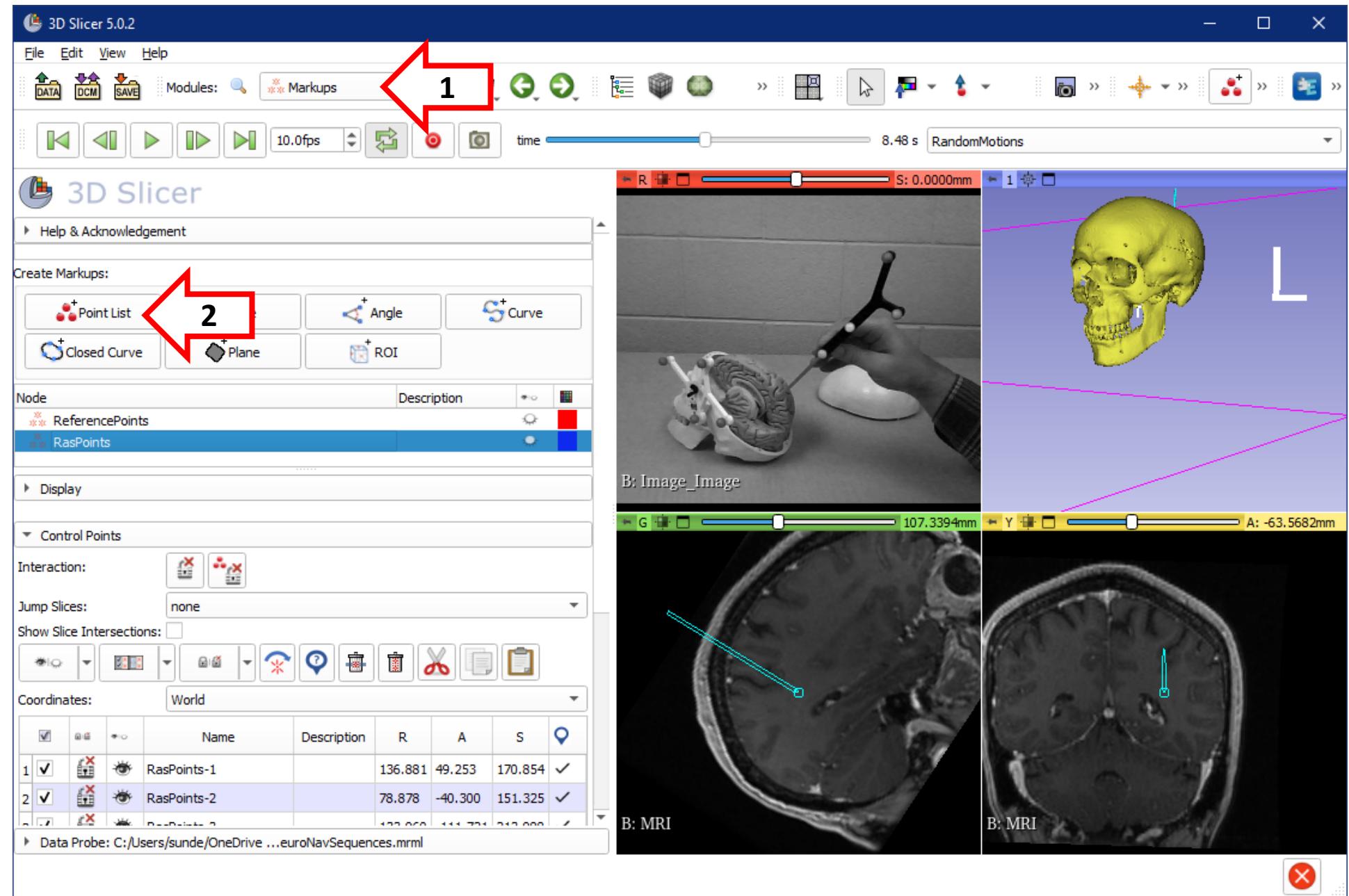


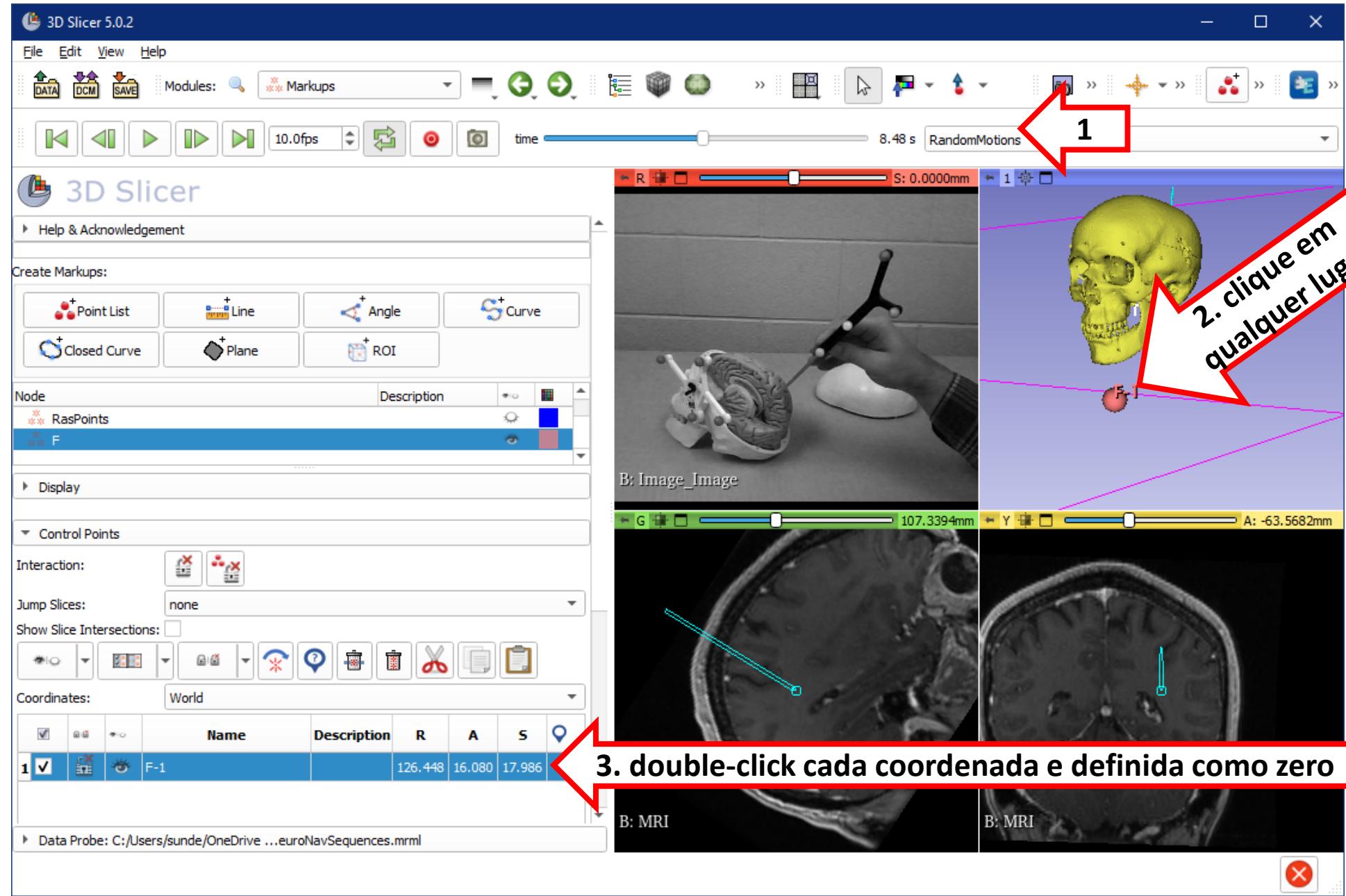


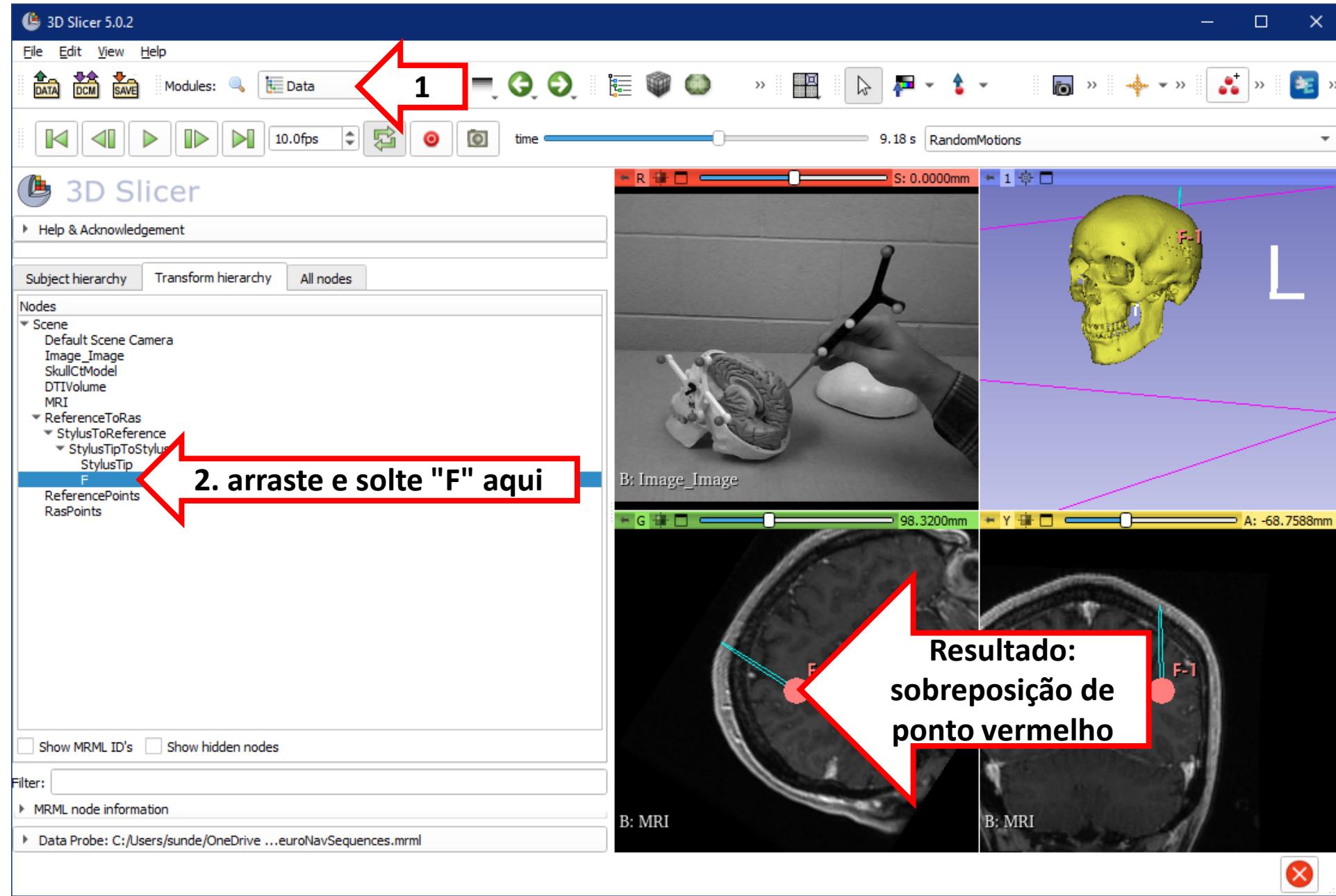


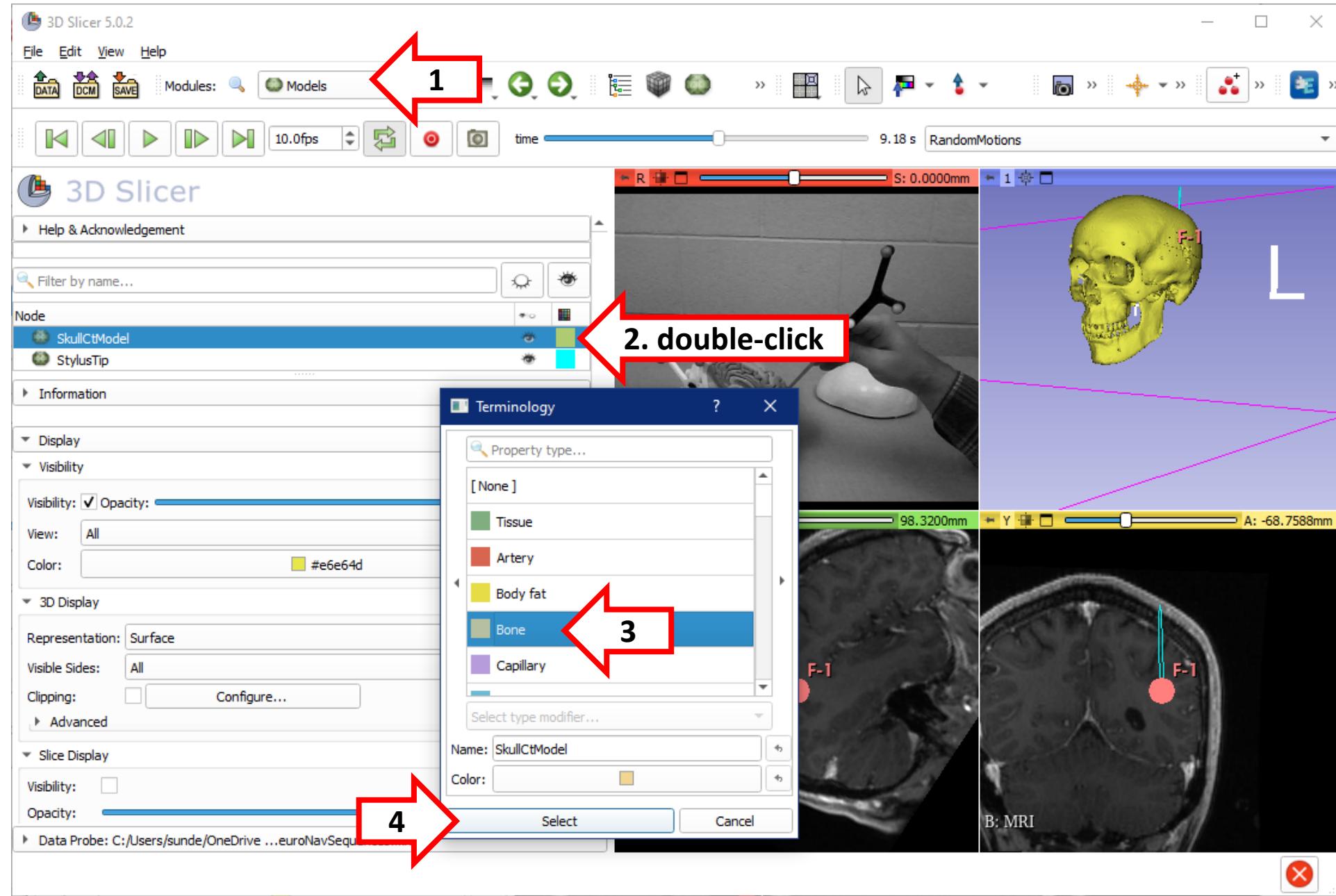


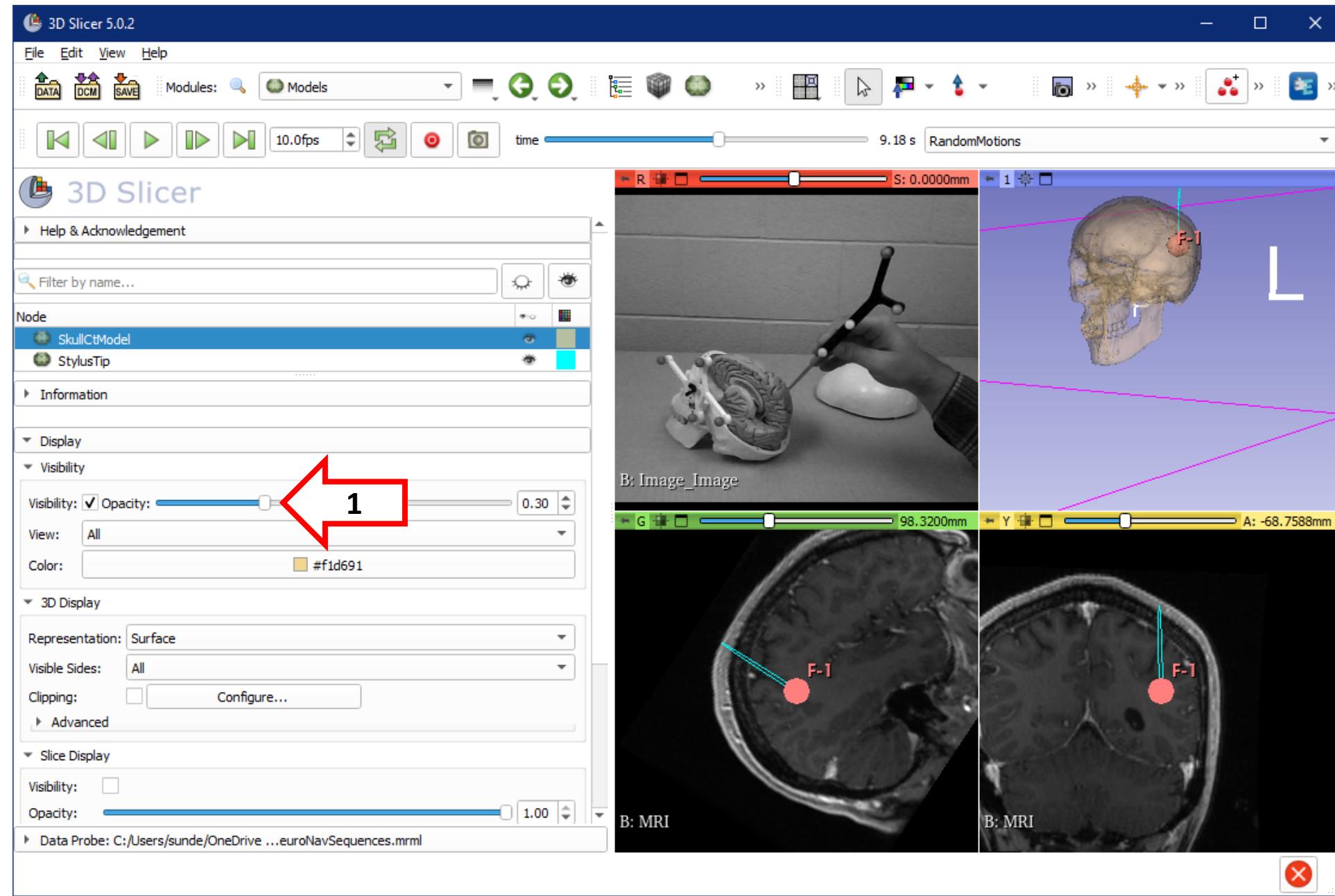


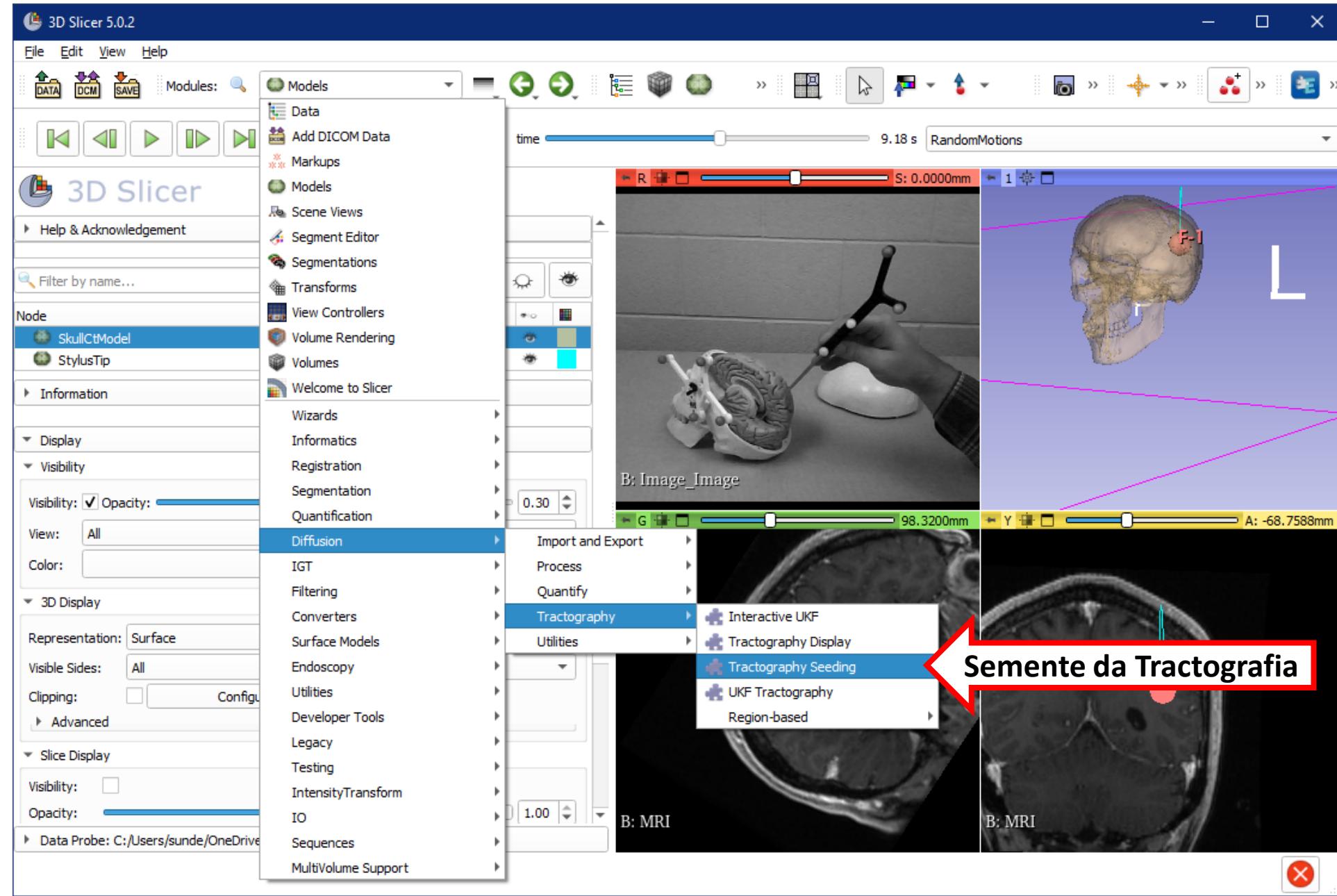


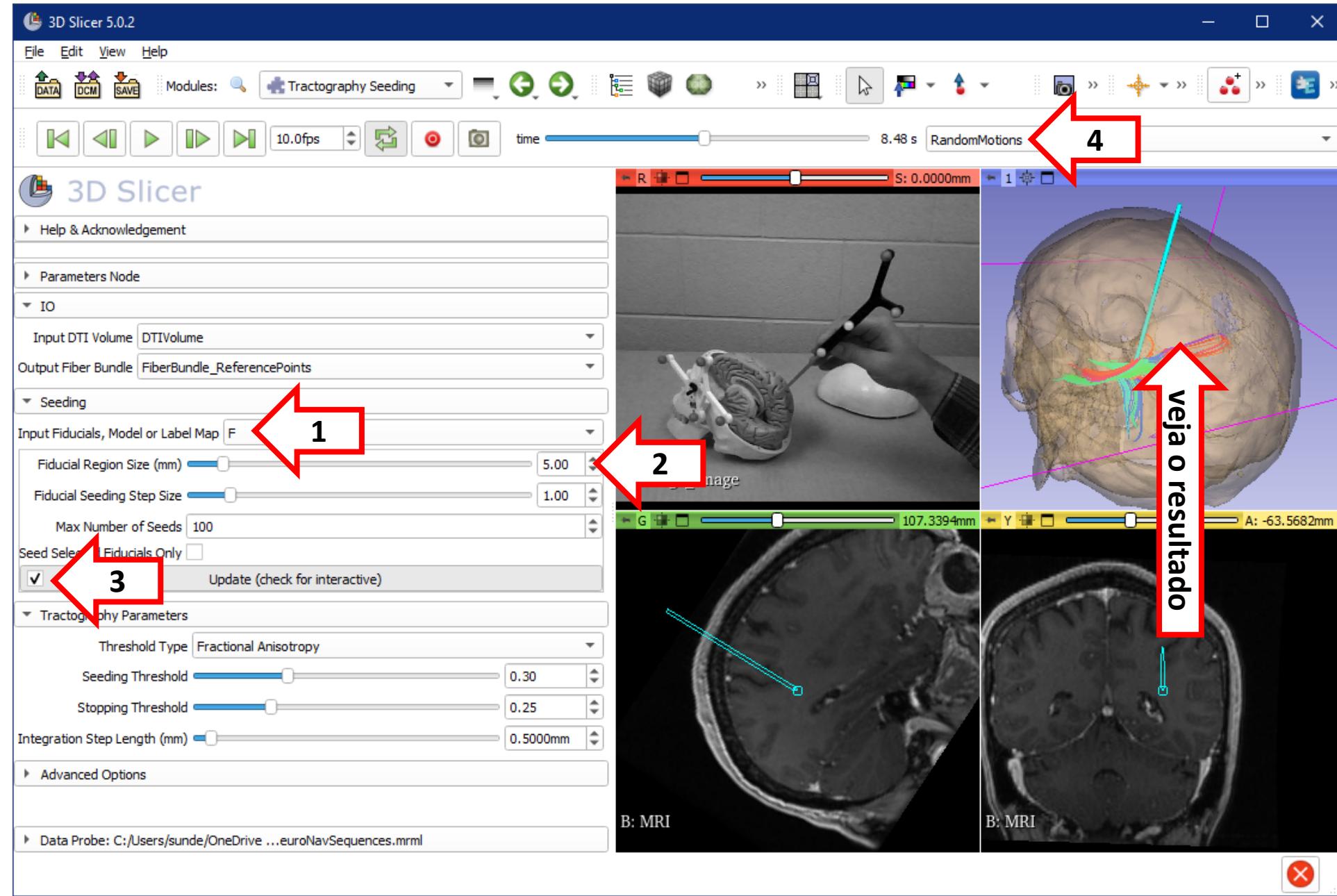












Programação

Parte 1

- Arquitetura do software

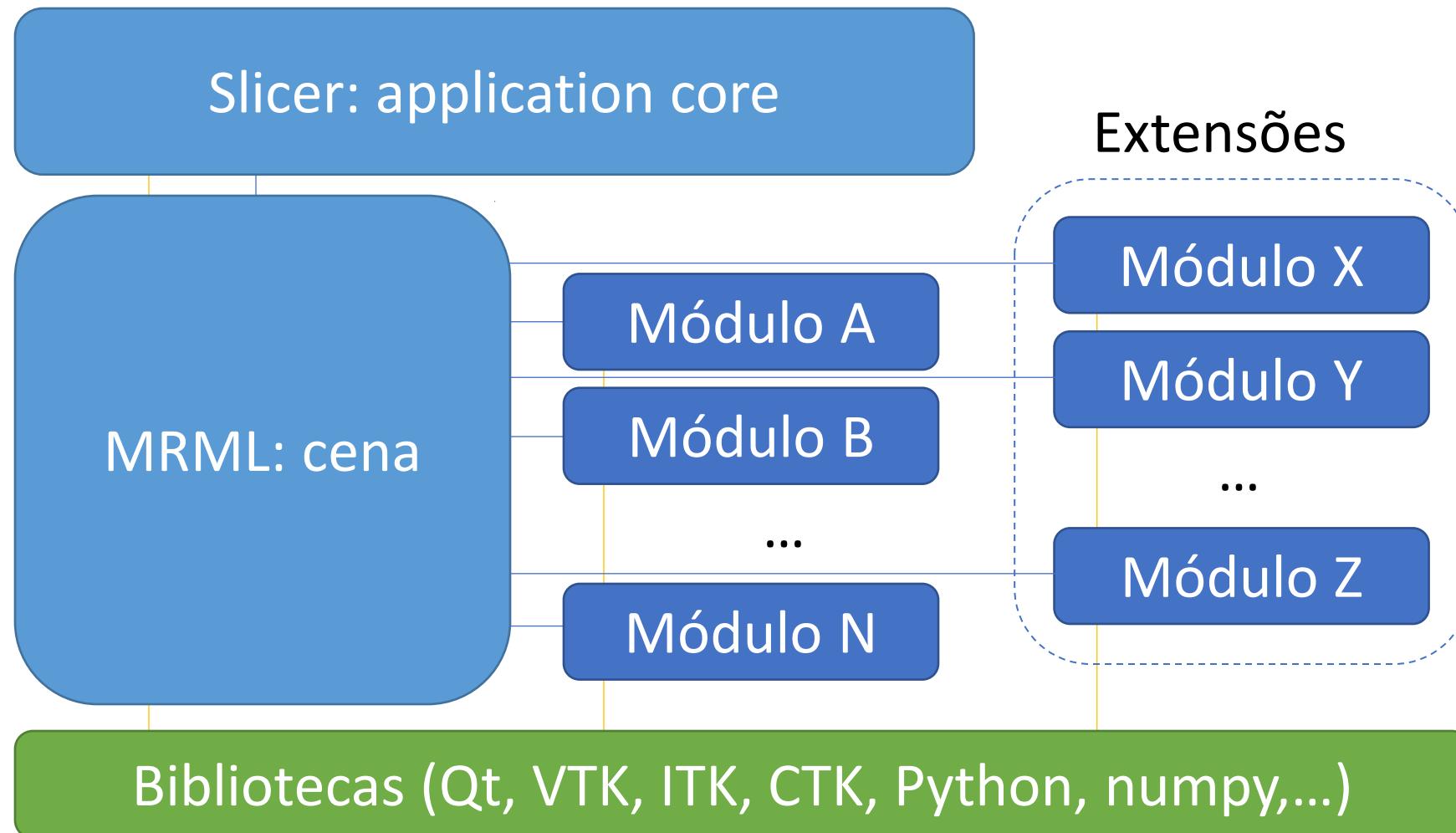
Parte 2

- Usar o console python no Slicer
- Exemplo de módulo com script simples

Parte 3

- Escreva um módulo com script simples individualmente

Arquitetura do aplicativo do Slicer



Tipo de módulo: C++ carregável

- Escrito em C++
- A API completa do 3DSlicer está acessível
- Útil para implementar componentes interativos complexos, críticos para o desempenho, infraestrutura de aplicativos (por exemplo, widgets GUI de baixo nível reutilizáveis)

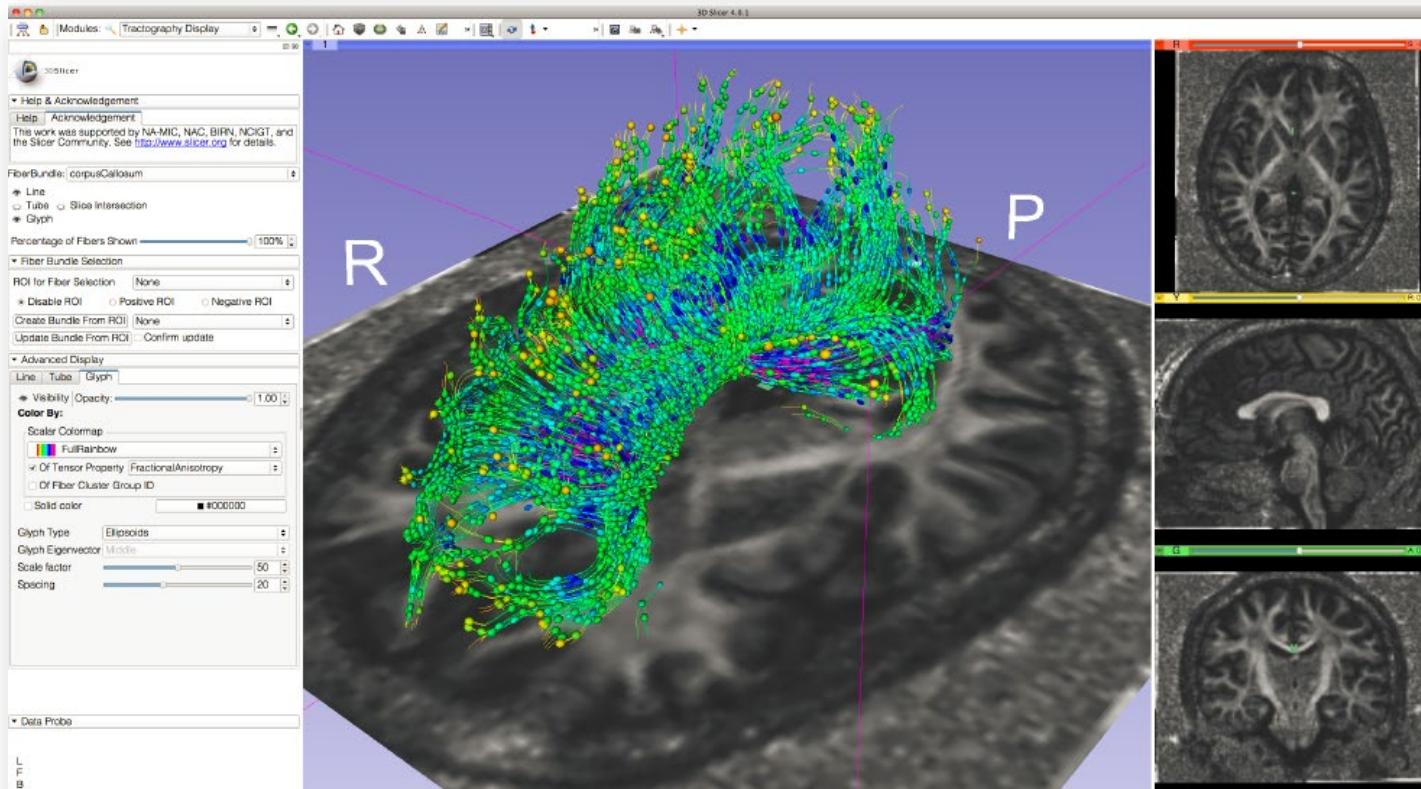
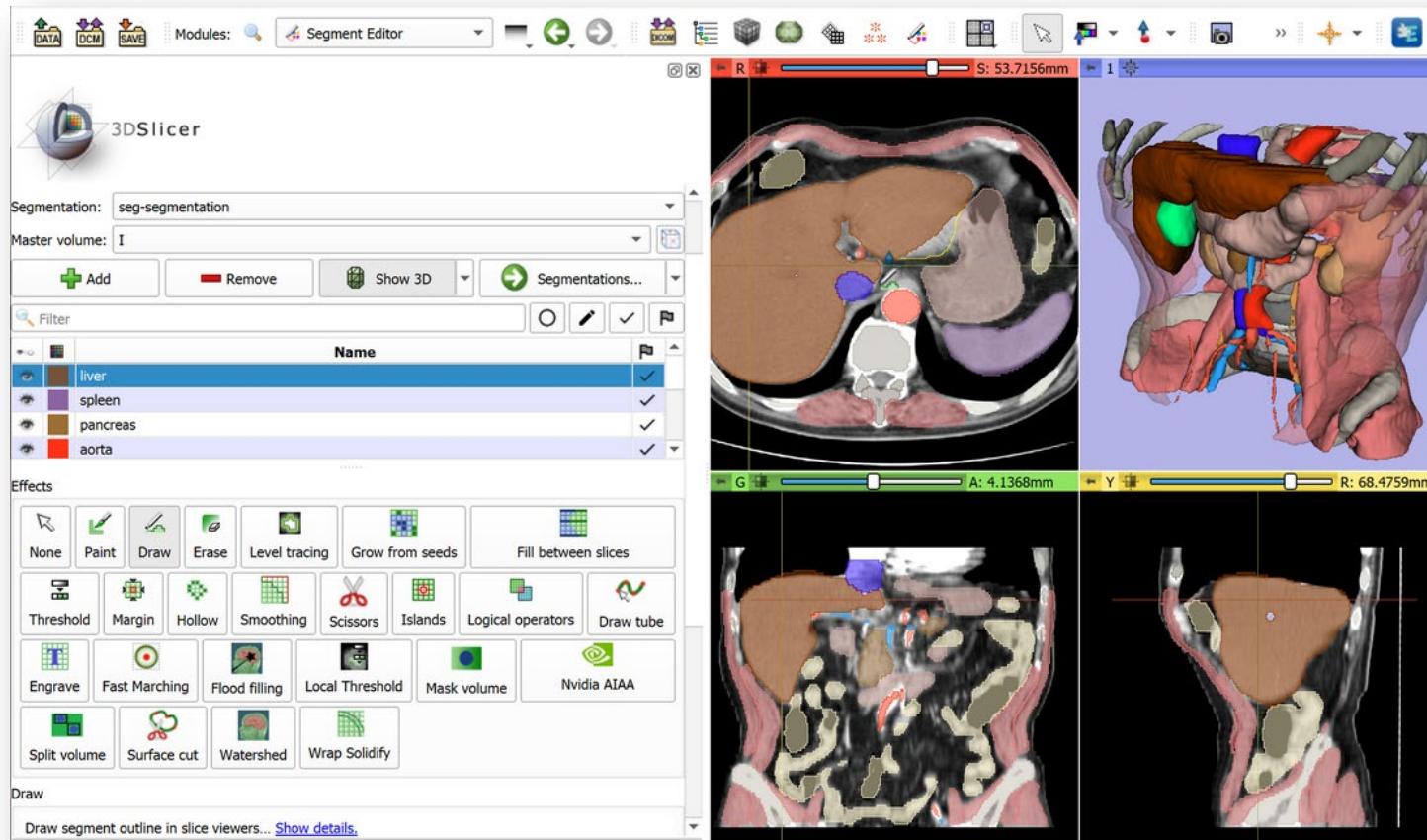


Imagen cortesia: Sonia Pujol

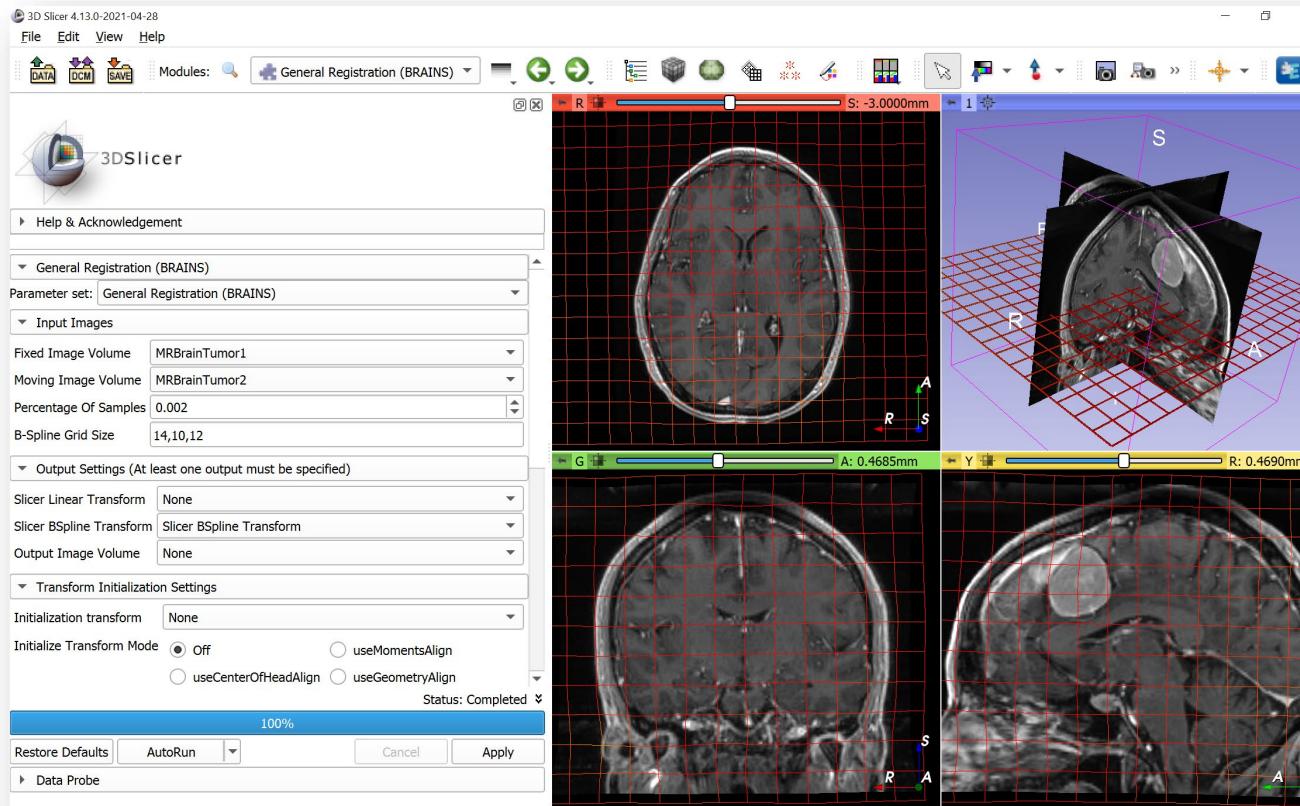
Tipo de módulos: Scripted carregável

- Escrito em Python
- A API completa do 3DSlicer está acessível
- Maneira mais simples de estender/personalizar Slicer



Tipo de módulo: CLI

- CLI = Interface de linha de comando
- Slicer pode executar qualquer aplicativo de linha de comando a partir da GUI
- Pode ser implementado em qualquer linguagem (C++, Python, ...)
- Entradas e saídas especificadas no arquivo XML, GUI é gerado automaticamente
- Bom para implementar algoritmos computacionais

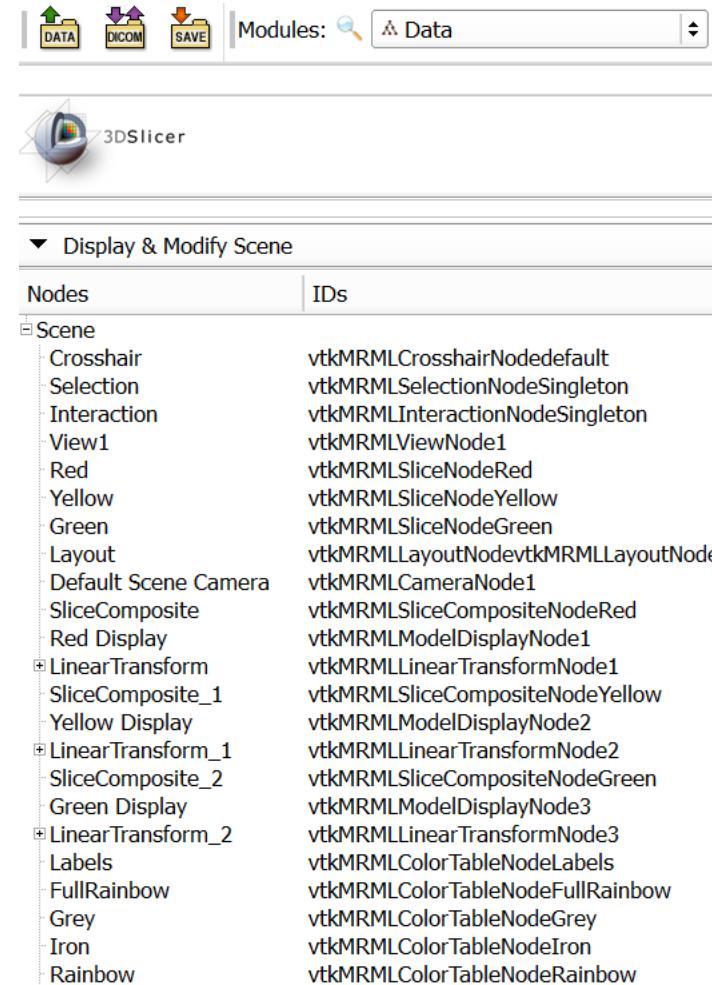


Modelo de dados do Slicer

- **Cena MRML:** armazenamento compartilhado de todos os objetos de dados

(MRML: Medical Reality Markup Language)

- Lista de nós MRML, cada um identificado por um ID de cadeia de caracteres exclusivo
- Referências, observações entre nós
- Os módulos se comunicam através da leitura/gravação de nós MRML
- Os módulos não precisam saber uns sobre os outros!



Nó MRML

- Responsabilidades:
 - Armazenar dados
 - Serialização de/para XML para armazenamento de arquivos
 - Sem exibição ou *métodos* processamento
- Tipos básicos:
 - Nό de dados
 - Nό de exibição: opções de visualização para o conteúdo do nό de dados; vários nós de exibição permitidos
 - Nό de armazenamento: qual formato, nome de arquivo usar para armazenamento persistente do conteúdo do nό de dados

Implementação de módulo com script

Módulo
(MyFirst)

Widget
(MyFirstWidget)

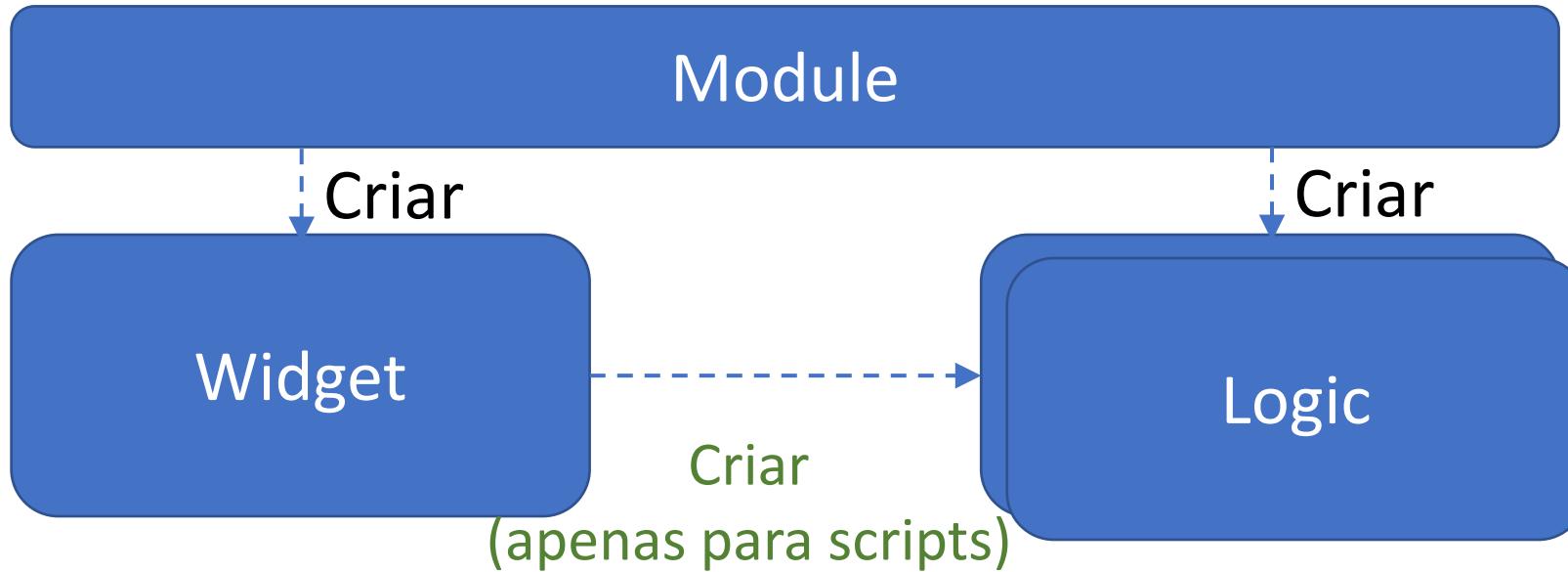
Lógica
(MyFirstLogic)

Classe Module

- Necessário. Existe apenas uma instância global:
`module = slicer.util.getModule('Volumes')`
- Armazena o nome do módulo, descrição, ícone, etc.
- Cria e mantém uma referência à lógica e ao widget:
`widget = slicer.util.getModuleWidget('Volumes')`
`logic = slicer.util.getModuleLogic('Volumes')`

- Em módulos carregáveis com script: `getModuleWidget()` requer que o widget tenha um membro lógico. Este é o caso do módulo com **lógica passiva**.
- Widget e lógica não têm muito uso para módulos CLI. Use-os via MRML (editar nó de parâmetro CLI) e `slicer.cli.runSync()`.

Implementação de módulo com script

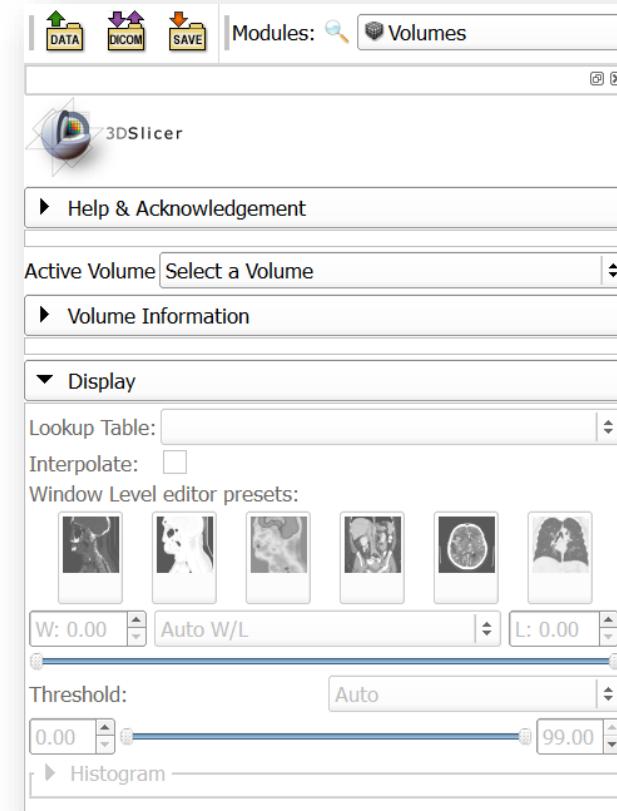


A lógica do módulo com script não é criada automaticamente (para facilitar o recarregamento dinâmico do módulo sem reiniciar o aplicativo). A lógica deve ser instanciada manualmente:

- Opção A: Lógica passiva. Crie um objeto lógico na classe Widget. Este é o mais simples e mais comumente usado (usado no modelo de módulo com script). Se outros módulos precisarem acessar a lógica, eles poderão instanciar a lógica do outro módulo (veja o exemplo).
- Opção B: Lógica ativa. Crie um objeto lógico no sinal startupCompleted() e armazene-o na classe do módulo. Por exemplo, este é o caso quando a lógica observa a cena e reage a mudanças, modifica nós automaticamente. Apenas uma dessas lógicas é permitida por aplicativo, porque várias lógicas podem interferir umas com as outras.

Classe Widget

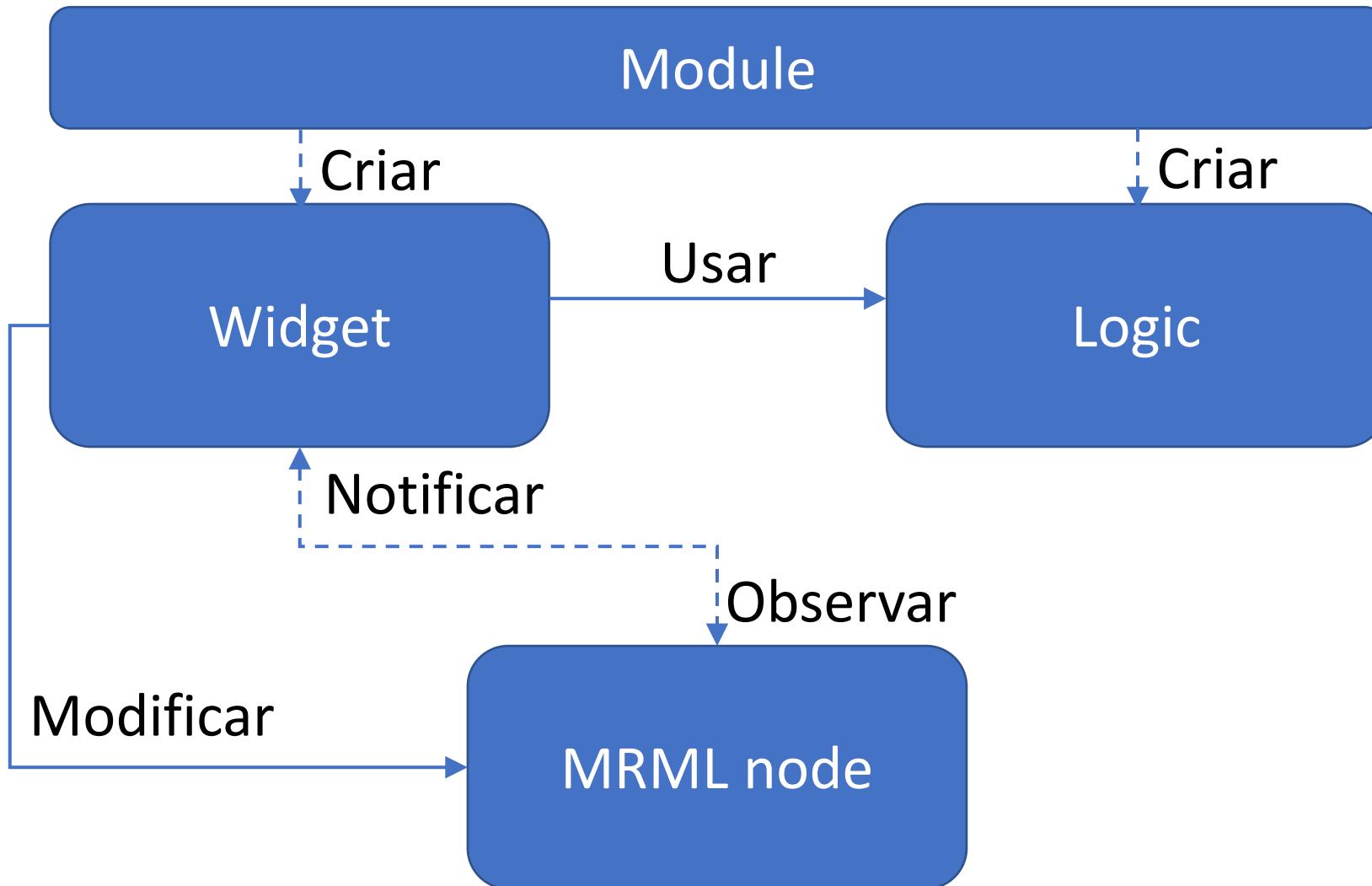
- Necessário se o módulo tiver uma interface de usuário
- Normalmente, existe apenas uma instância global
- Define a interface do usuário do módulo
- Mantém a interface do usuário e os nós sincronizados (observa os nós MRML para obter notificações de alteração)
- Inicia métodos de processamento implementados na classe lógica



Classe Widget

- Incluir um seletor de nó de parâmetro na parte superior (ou usar um nó de parâmetro singleton)
- Se um nó de parâmetro for selecionado, adicione um observador aos seus eventos modificados; se modificado, em seguida, chamar o método widget `updateGUIFromParameterNode()`
- Se o usuário alterar a GUI, atualize o nó MRML chamando métodos que atualizam valores no nó de parâmetro. Um compartilhadométodo `updateParameterNodeFromGUI()` também pode ser usado, o que é chamado em qualquer interação do usuário GUI.

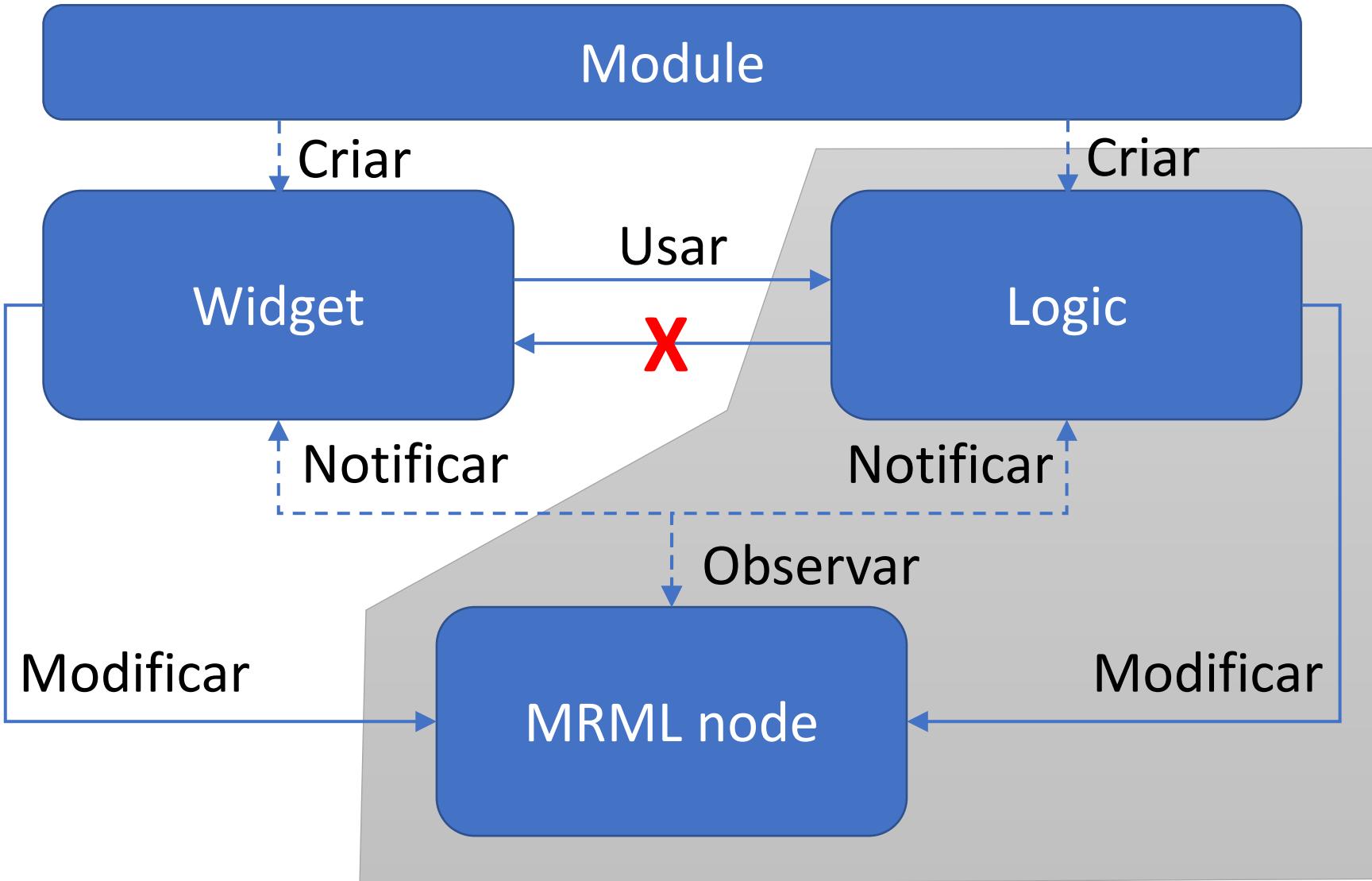
Implementação de módulo com script



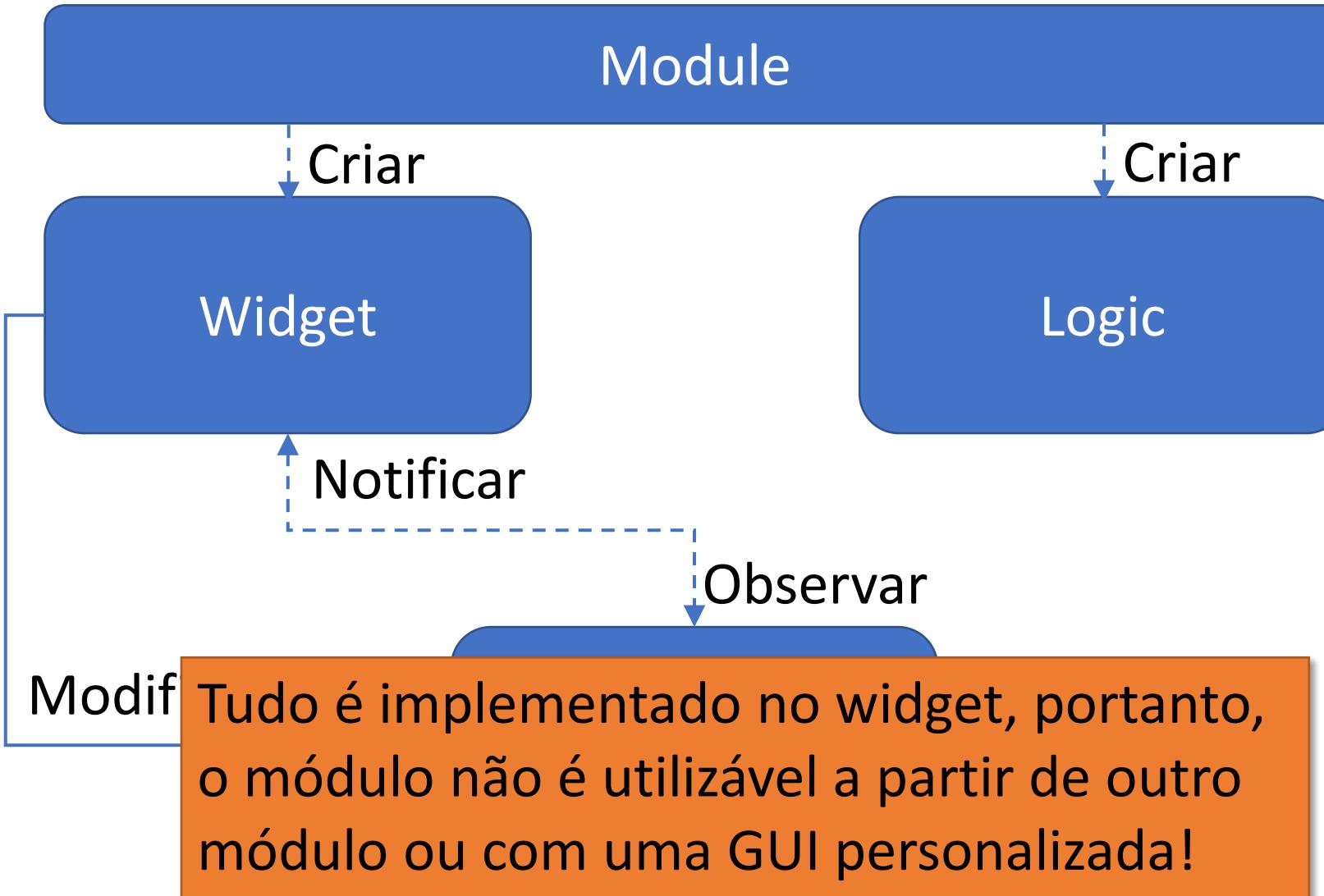
Classe lógica

- Necessário se o módulo fizer qualquer processamento (sempre)
- O módulo deve ser utilizável a partir de outro módulo, apenas chamando métodos lógicos
- Não deve depender da classe Widget: o módulo deve ser utilizável sem sequer ter uma GUI
- A lógica pode ser instanciada muitas vezes (para acessar funções do utilitário dentro)
- A lógica pode observar nós (lógica ativa): somente se o processamento em segundo plano em tempo real for necessário (por exemplo, observamos alguns nós de entrada e atualizamos outros nós se os nós de entrada forem alterados)

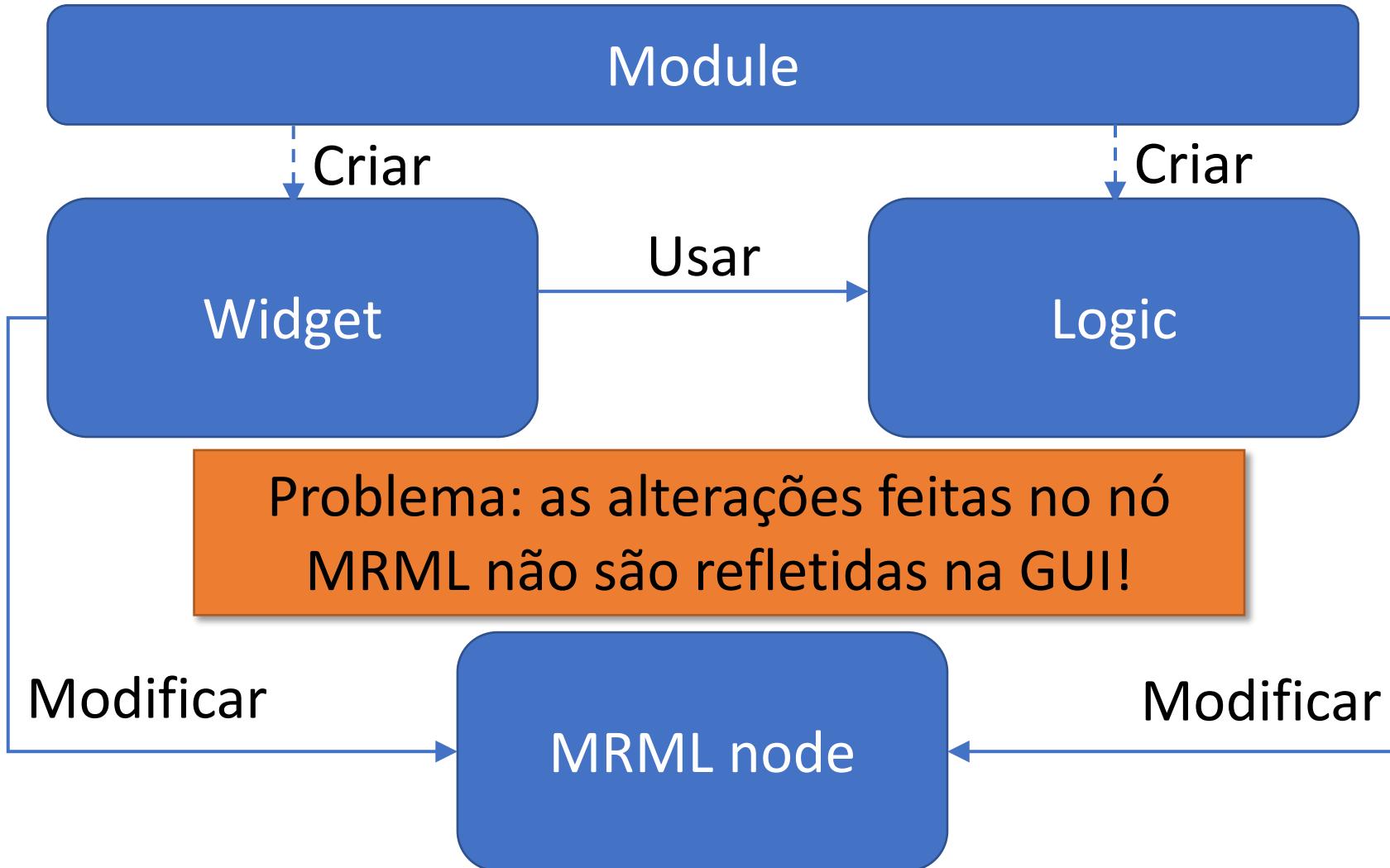
Implementação de módulo com script



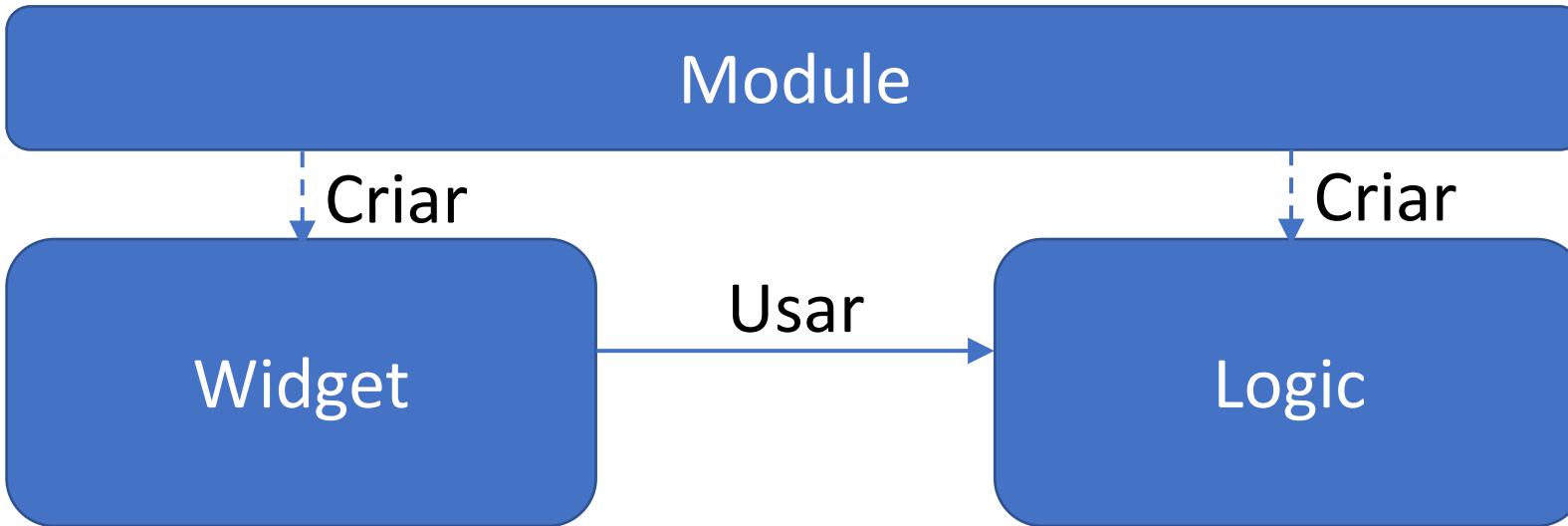
Erros comuns 1



Erros comuns 2

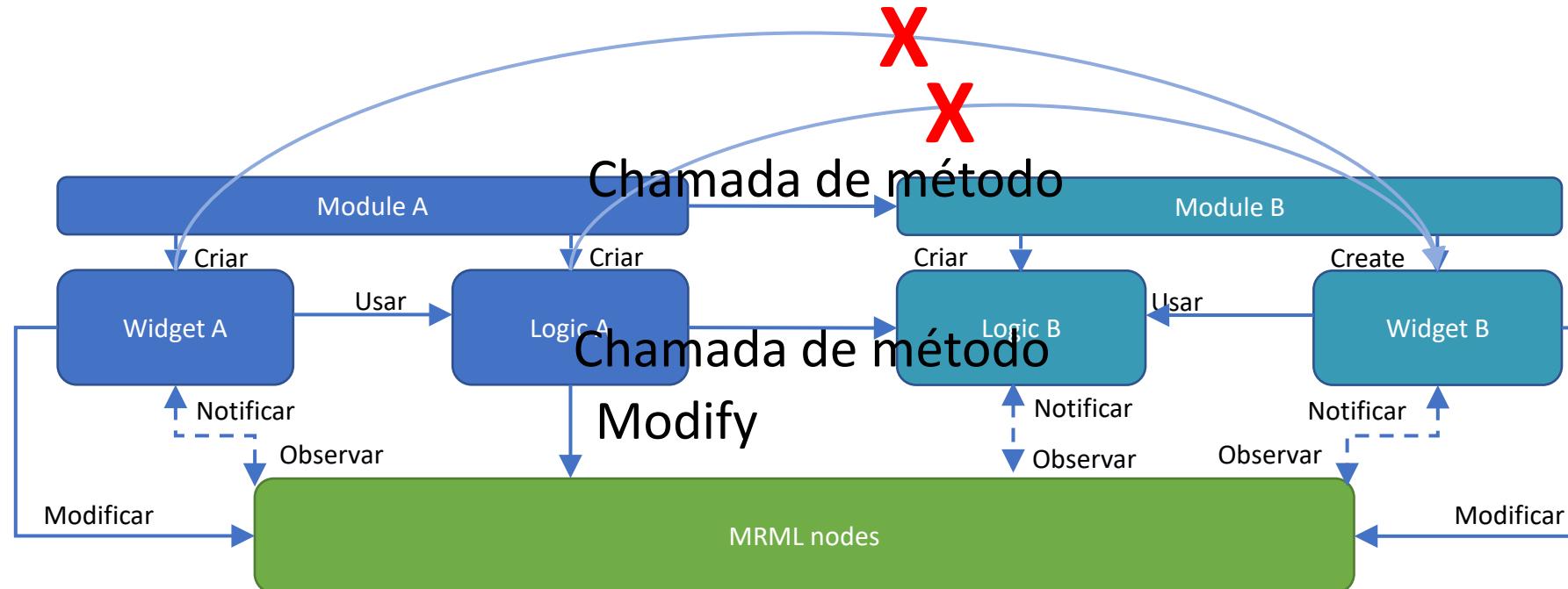


Erros comuns 3



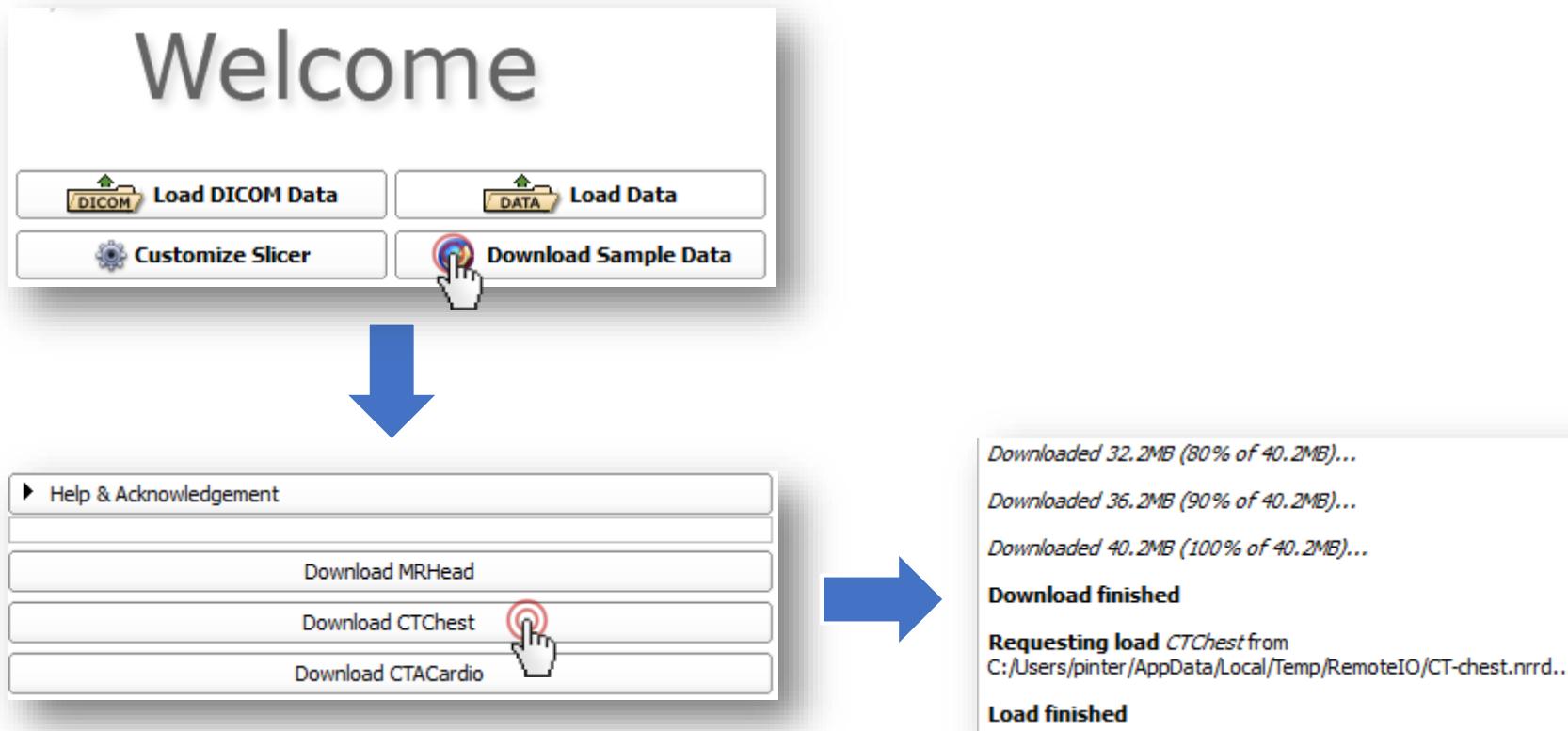
Nenhum nó de parâmetro é usado. Quando a cena é salva e recarregada, todas as configurações na interface do usuário do módulo são perdidas!

Como o módulo A pode usar o módulo B?

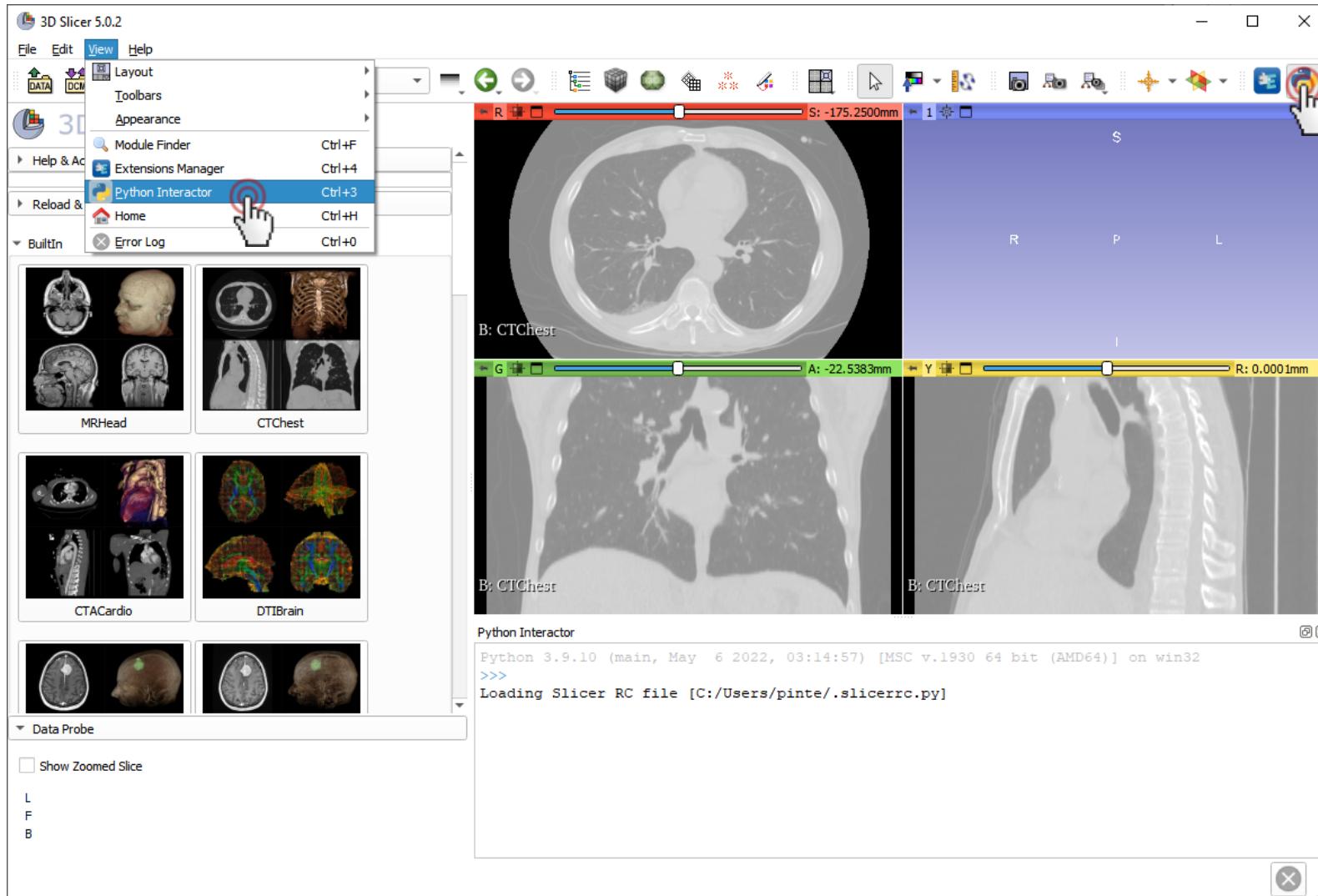


- A lógica do módulo pode modificar quaisquer nós MRML – forma mais comum de comunicação
- A classe lógica do módulo pode usar a classe lógica de outro módulo
- A classe Module pode usar outra classe de módulo, mas isso raramente é necessário (por exemplo, para acessar recursos que exigem Qt ou outras classes GUI)
-

Iniciar o Slicer e carregar dados

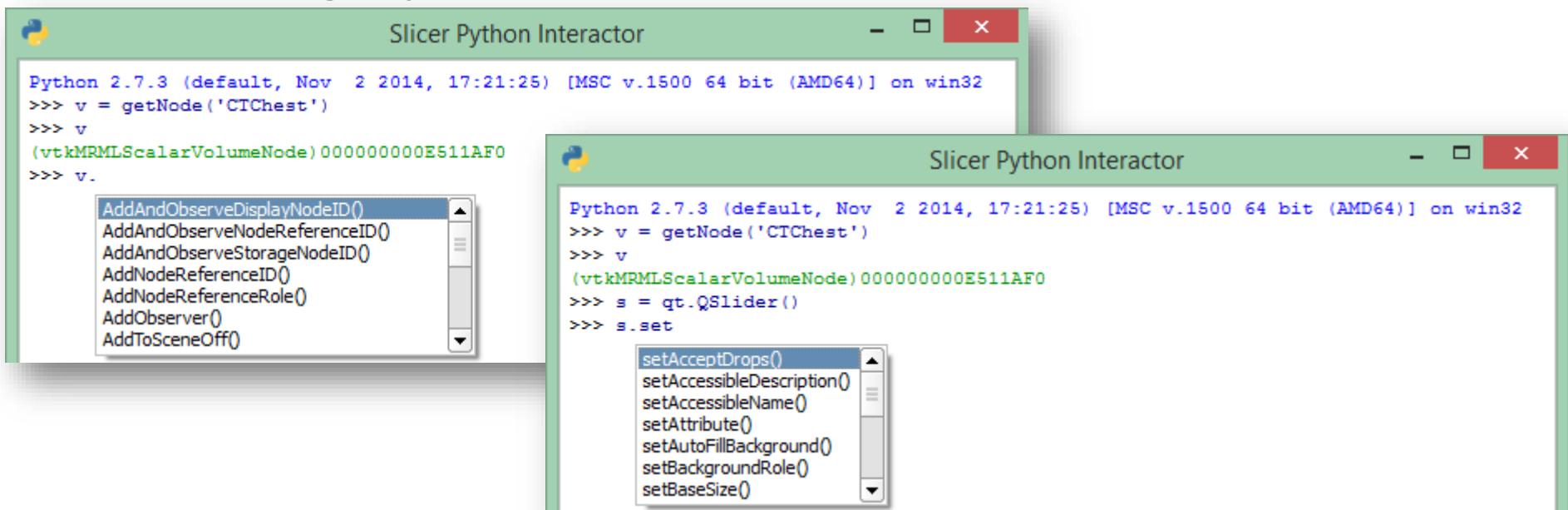


Apresentando o console python



Recurso de preenchimento automático

- Ferramenta essencial que fornece informações da API
- Pressione TAB para abrir a janela de preenchimento automático para
 - Explorar as funções disponíveis de um determinado objeto
 - Economizar digitação



Acessando a cena MRML e os nós

- Usando funções utilitárias – em `slicer.util`

```
v = getNode('CTChest')
```

OR

```
v = getNode('CT*')
```

`getNode`: um pouco ambíguo, recomendado apenas para testes e depuração

- Acessando a cena MRML diretamente

```
v=slicer.mrmlScene.GetFirstNodeByName('CTChest')
```

OU

```
v=slicer.mrmlScene.GetFirstNodeByClass('vtkMRMLScalarVolumeNode')
```

Informações sobre variáveis

- Obter tipo de variável e ponteiro: insira o nome da variável

v

(vtkMRMLScalarVolumeNode)0000008FF76243B8

Nota: É sempre bom verificar a variável depois de criá-la

- Mostrar conteúdo do nó: todos os membros e atributos da árvore de hierarquia

print(v)

- Mostrar API do nó: descrição de todos os métodos

help(v)

Manipulando Volumes

- Definindo valores de janela/nível programaticamente

```
vd = v.GetDisplayNode()
```

```
vd.SetAutoWindowLevel(0)
```

```
vd.SetWindowLevel(350,40)
```

- Que métodos/parâmetros estão disponíveis:

```
help(vd)
```

OU

```
vd
```

```
(vtkCommonCorePython.vtkMRMLScalarVolumeDisplayNode)0000021AD5436588
```

<http://www.slicer.org/doc/html/classes.html>

Manipulando volumes

- Acessando, alterando voxels – usando numpy
- Obtenha o valor do voxel em (100,200,30)

```
va = slicer.util.arrayFromVolume(v) <= obter voxels como uma matriz numpy
```

```
va[100,200,30]
```

-986 <= voxel value

- Thresholding

```
vaOriginal=va.copy() <= salvar os valores de voxel originais
```

```
va[v<200] = -3000
```

```
va[v>200] = 2000
```

```
slicer.util.arrayFromVolumeModified(v) <= indicar ao Slicer que as  
atualizações foram concluídas
```

- Processo com função arbitrária

```
va[:] = vaOriginal[:]*2.5-500; v.Modified()
```

Carregue o modelo

<http://perk-software.cs.queensu.ca/plus/doc/nightly/modelcatalog/>

Plus toolkit - printable 3D models catalog

Catalog of printable 3D models of tools and tracking fixtures.

Tools

See below a list of tools for tracking, calibration, and simulation.

Image	ID	Description	Printable model
	Scalpel	Generic scalpel (100mm long handle, 20mm long blade).	Scalpel.stl (2016-12-06-97ec12c) Source file(s)
	Cautery	Generic cautery (95mm long handle, 20mm long blade).	Cautery.stl (2016-12-06-97ec12c) Source file(s)

Manipulando modelos

- Definindo a cor do modelo programaticamente

```
c = getNode('Cautery')
cd = c.GetDisplayNode()
cdSetColor(1,0,0)
```

- Alterar modelo para uma esfera

```
s = vtk.vtkSphereSource()
s.SetRadius(30)
s.SetCenter(30,40,60)
s.Update()
c.SetAndObservePolyData(s.GetOutput())
```

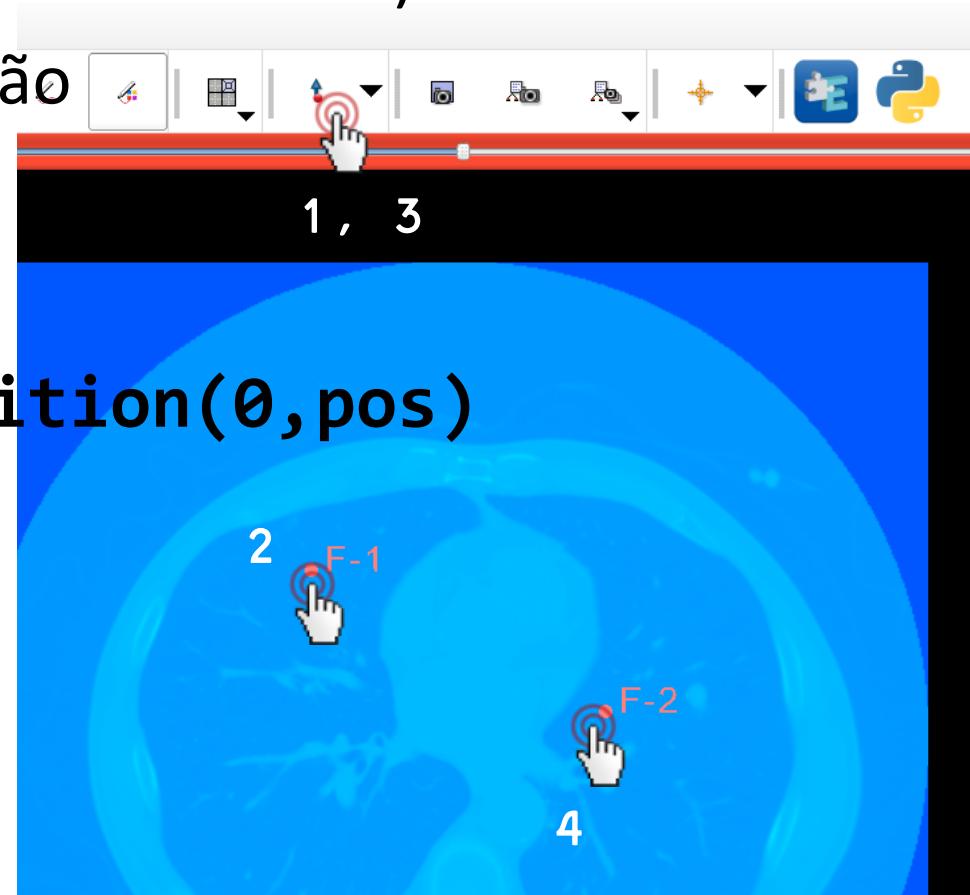
Manipulando marcações

- Criar 2 pontos de marcação (isso cria um nó de lista fiducial de marcação chamado 'F')
- Obter posição de marcação

```
f = getNode('F')  
pos=[0,0,0]  
f.GetNthFiducialPosition(0,pos)
```

pos

```
[58.93727622783058,  
 45.58082600473318,  
 -170.2500000000001]
```



Observando objetos MRML

```
def printPos(caller=None, event=None):
    f = getNode('F')
    pos=[0,0,0]
    f.GetNthFiducialPosition(0,pos)
    print(pos)
```

} These lines start with 2 spaces!

```
printPos()
```

```
[58.93727622783058, 45.58082600473318, -170.2500000000001]
```

```
obsTag=f.AddObserver(slicer.vtkMRMLMarkupsNode.PointModifiedEvent
, printPos)
```

=> Drag-and-drop first fiducial

```
[20.267904116373643, 4.977985287703447, -170.2500000000001]
[21.92516292115039, 1.6634676781499849, -170.2500000000001]
[22.477582522742637, 1.1110480765577364, -170.2500000000001]
[24.687260929111574, 0.5586284749655164, -170.2500000000001]
[26.34451973388832, 0.00620887337326792, -170.2500000000001]
```

```
f.RemoveObserver(obsTag)
```

Compartilhe suas ferramentas

- No espírito do paradigma de código aberto, é encorajado a compartilhar suas ferramentas
- As extensões compartilhadas
 - aparecem no Extension Manager
 - são testados todas as noites nas plataformas Slicer Factory
- Como compartilhar?
 - Ramifique ExtensionIndex do GitHub e suba a descrição da sua extensão em arquivo (.s4ext)
<https://github.com/Slicer/ExtensionsIndex>
 - Peça à equipe principal para integrar (envie um “pull request”)

Parte 2

- Usar o console python no Slicer
- Exemplo de módulo com script simples

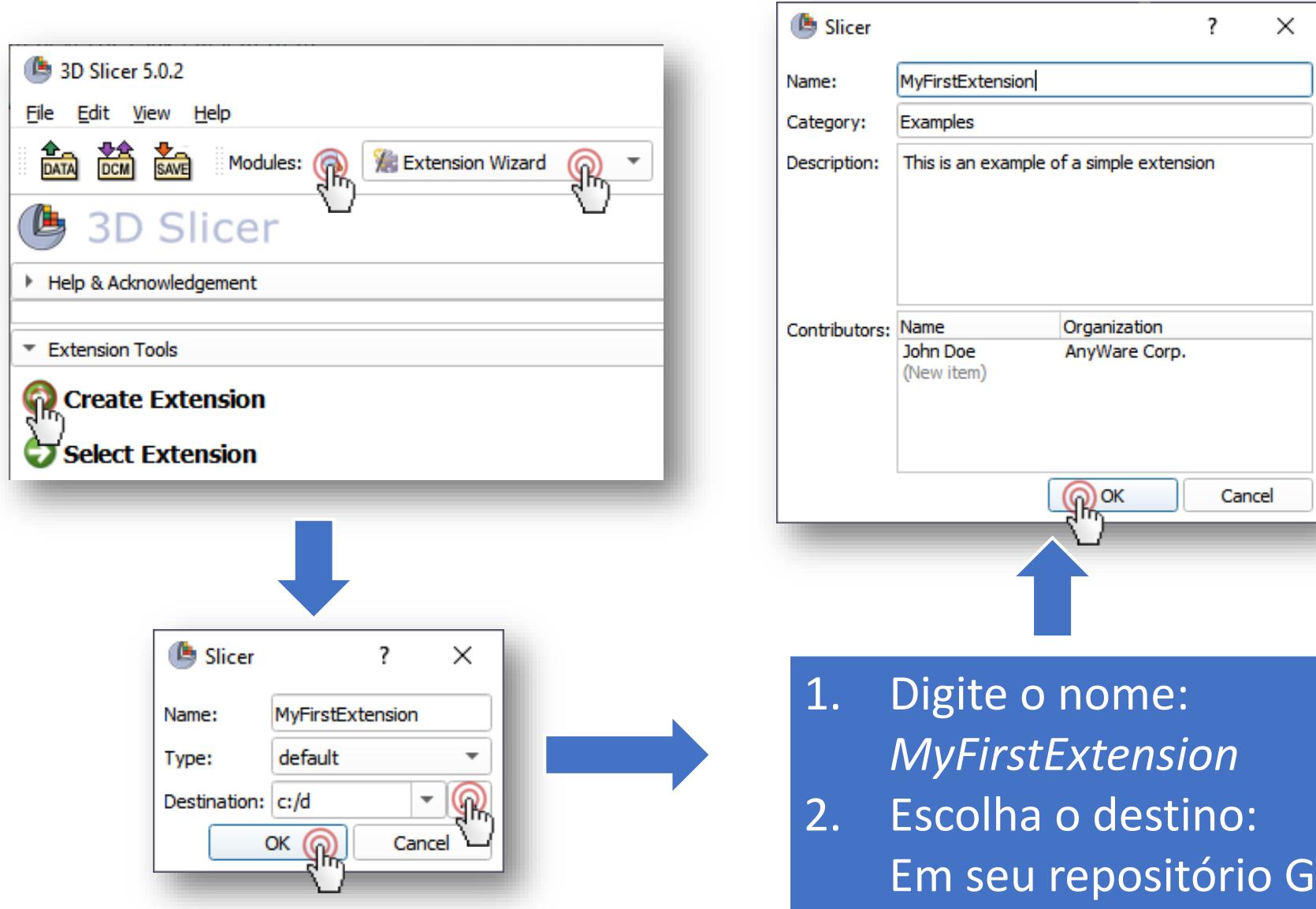
Python em geral

- **Blocos definidos por identação: 2 Espaços**
- Diferencia maiúsculas
- Comentários
 - # This whole row is a comment
 - """This is a potentially multi-line comment"""
- Blocos definidos por indentação
- Um objeto refere-se a si mesmo como `self` (em C++: `this`)
- Namespaces: `slicer`, `ctk`, `vtk`, `qt`
- Blocos... indentação... Espaços!

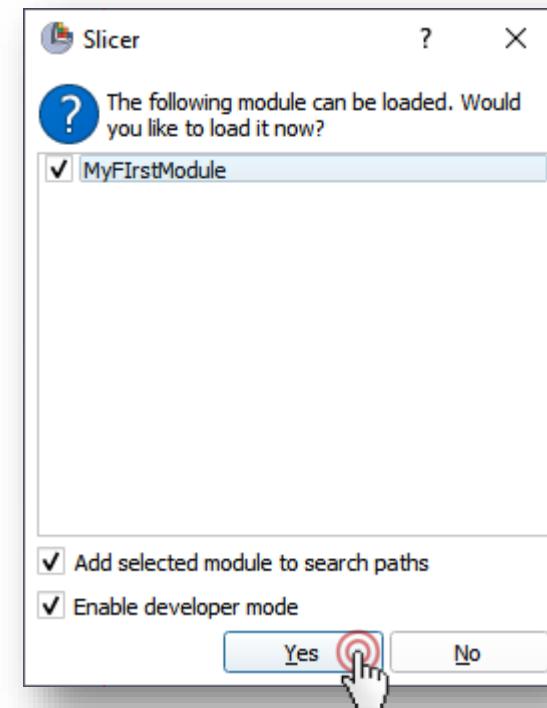
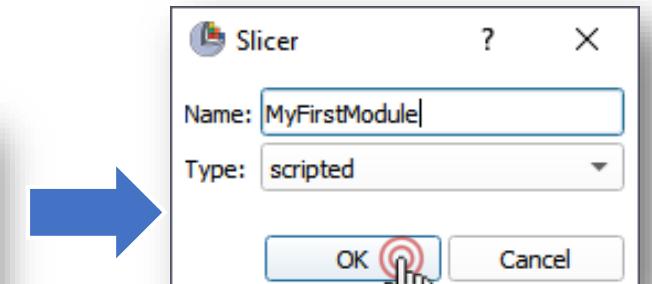
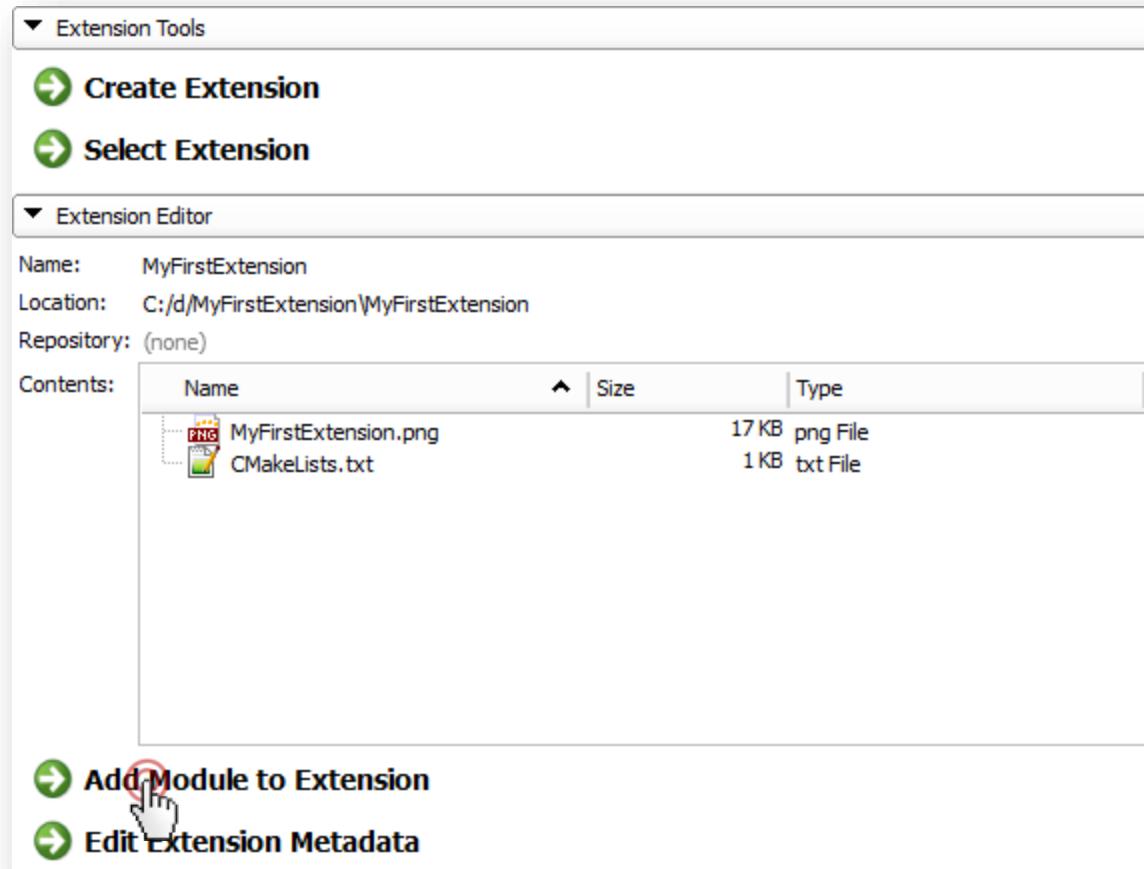
Editor de texto / IDE

- Usar um editor de texto adequado é essencial
- “Substitua tudo”, Fácil comentar/descomentar, identar, ...
- Realce de sintaxe
- Atalhos de teclado
- Recomendado:
Cross-platform: [**Visual Studio Code**](#), [**Sublime Text**](#)
 - Somente Windows: [Notepad++](#), Somente para Mac: Xcode
- Ambiente de desenvolvimento integrado - IDE:
 - Text editor + debugger, code browser, ...
 - Recomendado: [**PyCharm**](#), [**Visual Studio Code**](#), [**LiClipse**](#)

Criar extensão

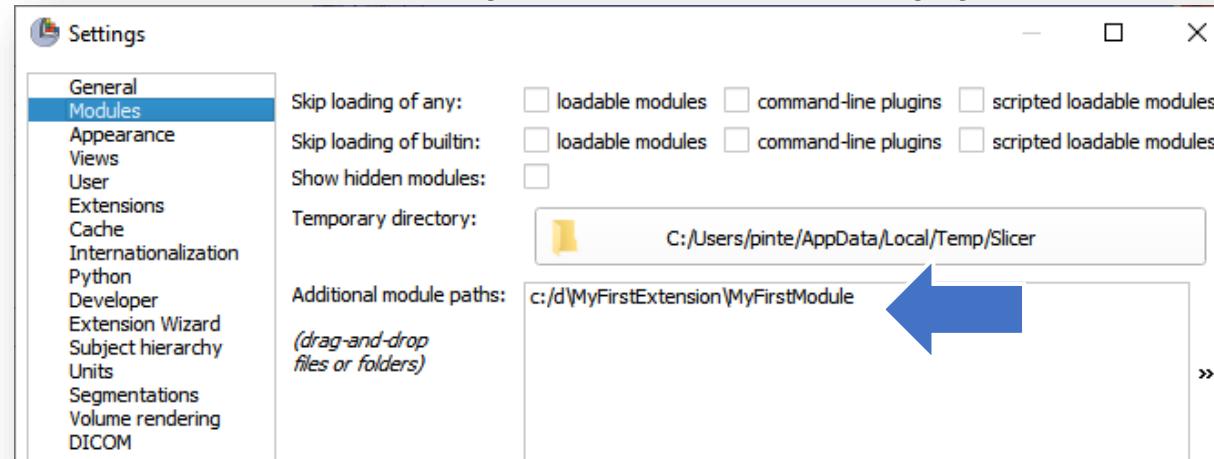


Criar módulo



Caminhos do módulo

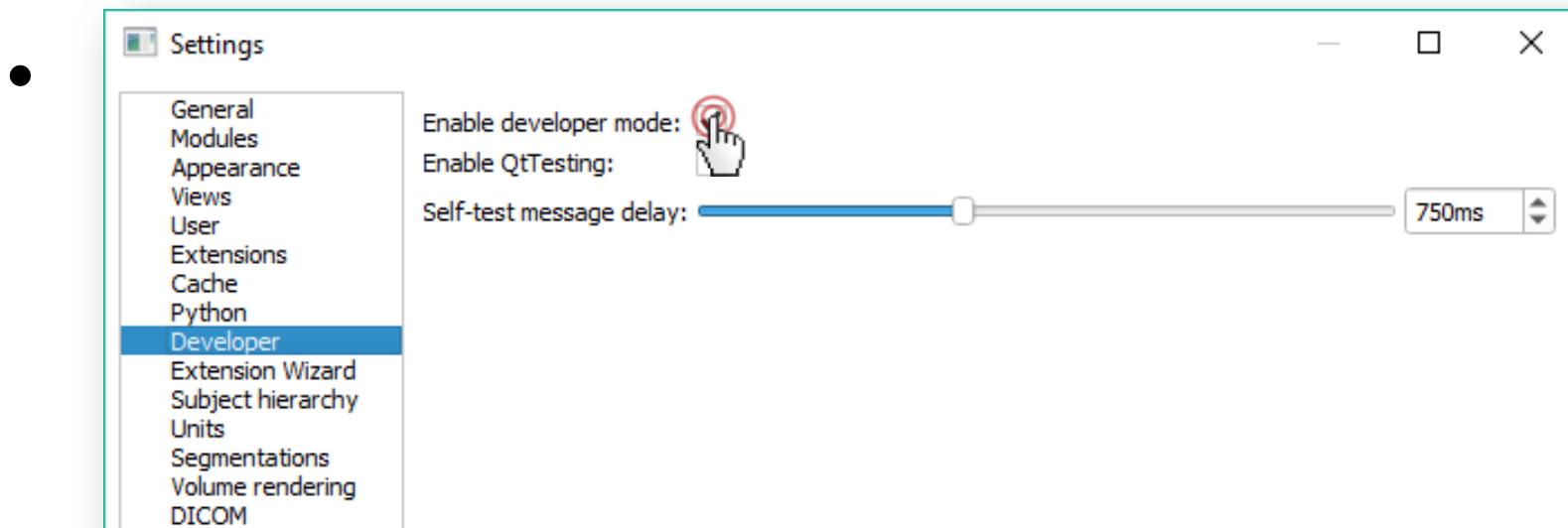
- Para fazer com que a segmentação de dados carregue um módulo, adicione sua pasta (em que o .py file está localizado) para “Additional module paths” em Application Settings



- Você pode arrastar e soltar o arquivo .py ou pasta
- Verifique se o checkbox “Add selected module to search paths” já adicionou os módulos listados a esta lista de caminhos

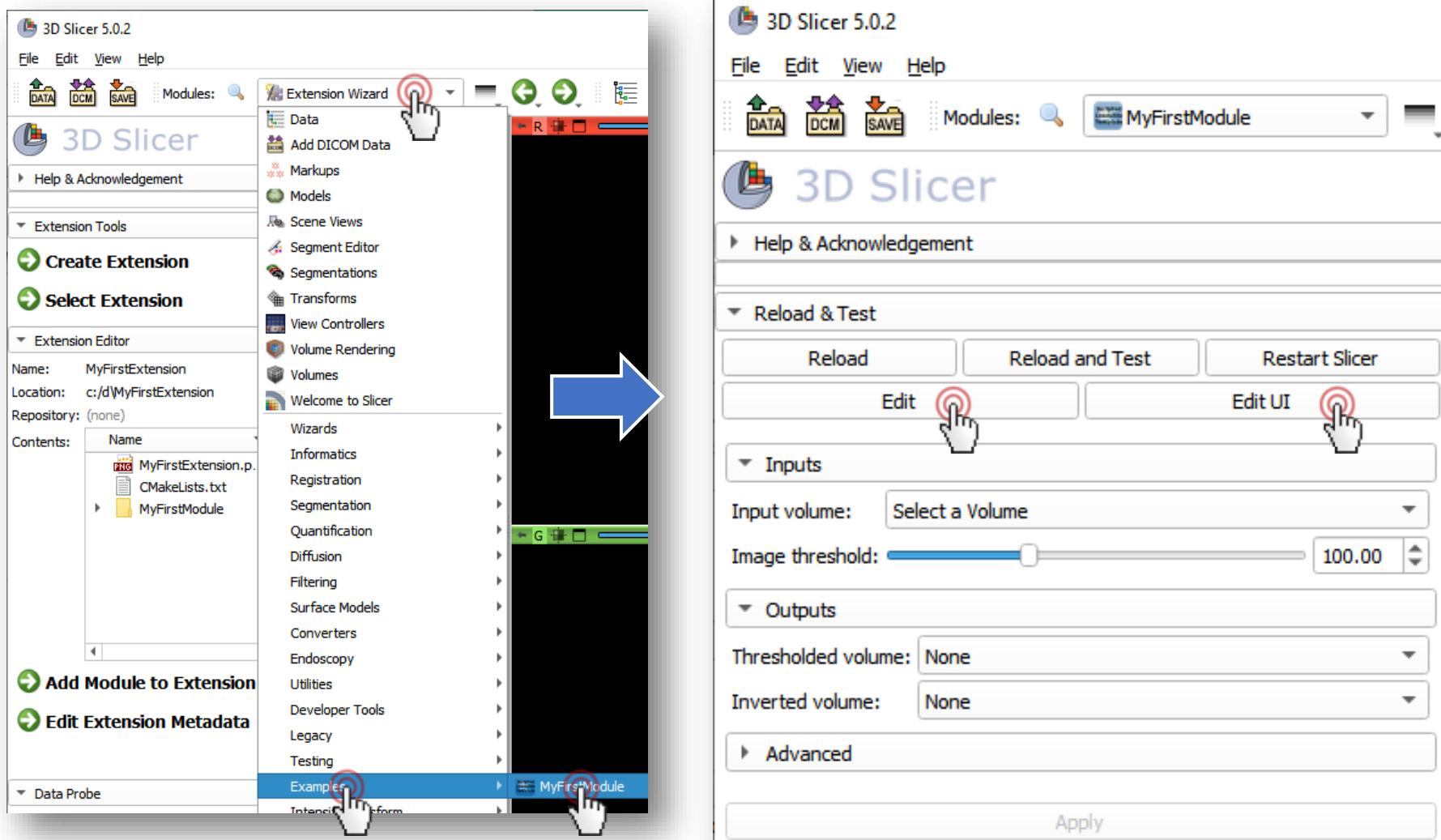
Modo de desenvolvedor

- Habilitar recursos úteis para o desenvolvimento
- Permitir a recarga dinâmica do código-fonte, etc..
- Em Configurações do aplicativo



- Reiniciar Slicer

Encontre o novo módulo no Slicer



“Commit” suas alterações regularmente

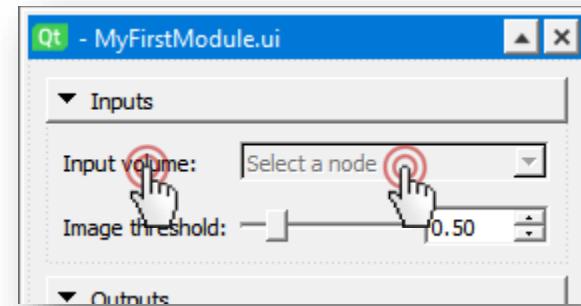
- Commit suas alterações
- Novos arquivos precisam ser adicionados explicitamente
- A mensagem de commit deve ter esta aparência:
"Re #7: Descrição de por que eu fiz o que eu fiz"
(não descrever o que você fez, é óbvio a partir do diff)
- Quando você acha que terminou
"Test # 7: Descrição de por que eu fiz o que eu fiz"
- When everybody agrees you're done
"Fixed #7: Descrição de por que eu fiz o que eu fiz"
- Pull (Fetch+Rebase) antes de cada commit!

Escreva nosso módulo com script #1

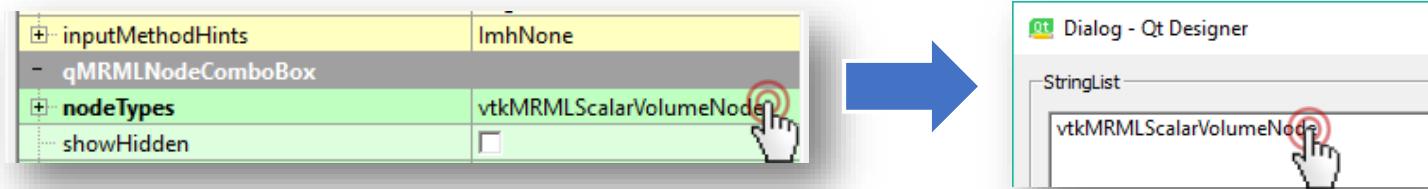
- Renomeie o módulo no arquivo .py
- ```
def __init__(self, parent):
 ScriptedLoadableModule.__init__(self, parent)
 self.parent.title = "Center of Mass"
```

# Escreva nosso módulo com script #2

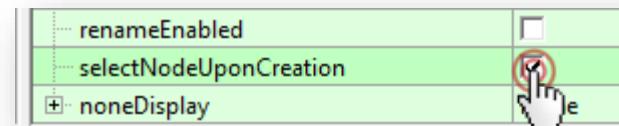
- Personalizar widgets no Qt Designer:
  - Double-click “Input Volume” e digite “Input fiducials:”
  - Selecione o seletor de nós ao lado de ‘Input fiducials’, e editar nodeTypes



- Double-click em “vtkMRMLScalarVolumeNode” e substitua-o por “**vtkMRMLMarkupsFiducialNode**”
- Desativar “selectNodeUponCreation”

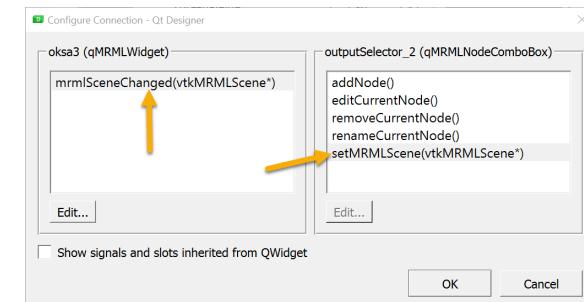
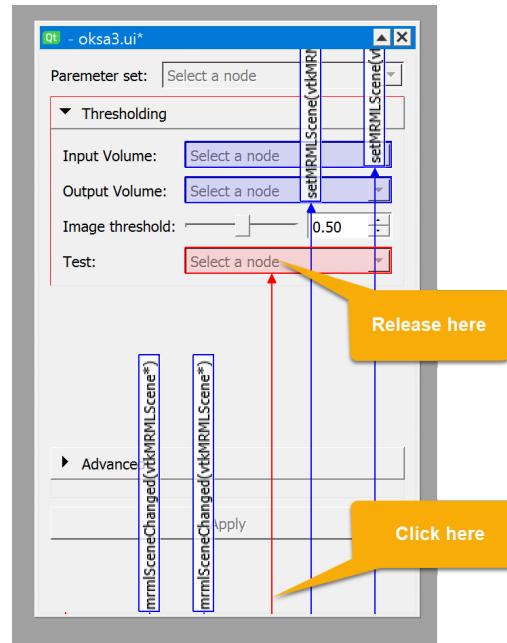


- Não se esqueça de salvar o arquivo.ui em Qt Designer



# Escreva nosso módulo com script #3

- Widgets MRML, como qMRMLNodeComboBox requerem a configuração da cena MRML
- A cena pode ser definida no Designer conectando o sinal ***mrrlSceneChanged(vtkMRMLScene\*)*** do widget de nível superior para o slot ***setMRMLScene(vtkMRMLScene\*)*** do widget
  - Opção A: Usar o Editor de Sinais/Slots (por padrão no canto inferior direito)
  - Opção B: Mude para o modo Editar Sinais/Slots (menu: Editar -> Editar Sinais/Slots ou tecla F4) e pressione o botão do mouse em uma área vazia no widget de nível superior, mova o mouse sobre o widget MRML e solte o botão do mouse; em seguida, selecione o sinal e o slot



# Escreva nosso módulo com script #4

- Procurar class MyFirstModuleLogic
- Insira este código acima da função process

```
def getCenterOfMass(self, markupsNode):
 centerOfMass = [0,0,0]

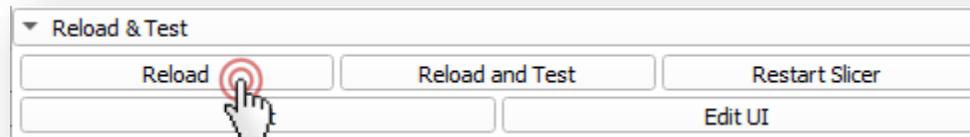
 import numpy as np
 sumPos = np.zeros(3)
 for i in range(markupsNode.GetNumberOfMarkups()):
 pos = np.zeros(3)
 markupsNode.GetNthFiducialPosition(i,pos)
 sumPos += pos

 centerOfMass = sumPos / markupsNode.GetNumberOfMarkups()

 logging.info('Center of mass for \'' + markupsNode.GetName() + '\': ' + repr(centerOfMass))

 return centerOfMass
```

- Clique no botão Recarregar para ver as alterações  
(lembre-se de salvar os arquivos primeiro)
- 



# Escreva nosso módulo com script #5

- Altere a função process para se parecer com isso:

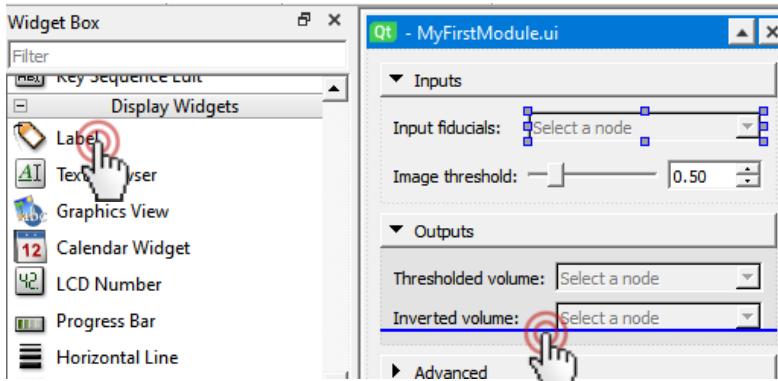
```
def process(self, inputMarkups, outputVolume, imageThreshold, enableScreenshots=0):
 """
 Run the actual algorithm
 """
 self.centerOfMass = self.getCenterOfMass(inputMarkups)

 return True
```

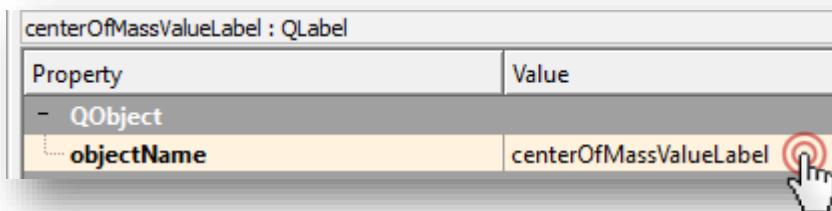
# Escreva nosso módulo com script #6

- Arraste e solte um Rótulo no widget do módulo na parte inferior da seção de saídas

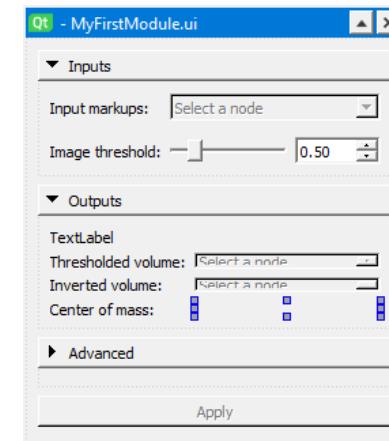
- 



- Clique duas vezes no rótulo e insira o “Center of mass:”
- Arraste e solte um segundo na coluna da direita, deixe-o vazio
- Renomeie o objeto para **centerOfMassValueLabel**



- Deve ser assim:



# Escreva nosso módulo com script #7

- Validando o estado do botão e exibindo a saída em **MyFirstModuleWidget – replace these functions:**

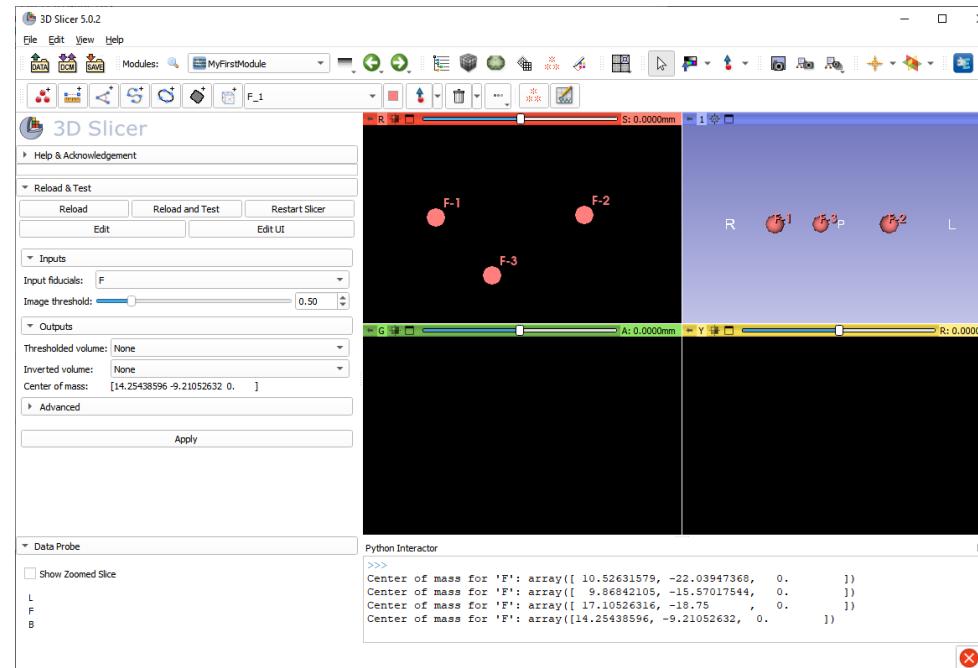
```
def updateGUIFromParameterNode(self, caller=None, event=None):
 if self._parameterNode is None:
 return
 self.ui.applyButton.enabled = self._parameterNode.GetNodeReference("InputVolume")

def onApplyButton(self):
 self.logic.process(self.ui.inputSelector.currentNode(),
 self.ui.invertedOutputSelector.currentNode(),
 self.ui.imageThresholdSliderWidget.value,
 not self.ui.invertOutputCheckBox.checked)
 self.ui.centerOfMassValueLabel.text = str(self.logic.centerOfMass)
```

- Preste atenção à identação correta!

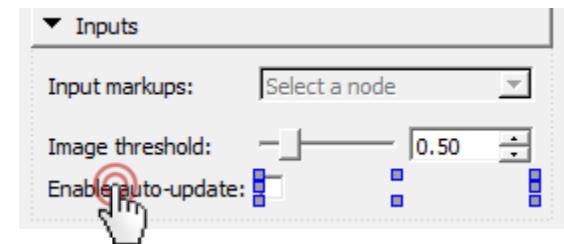
# Experimente o nosso módulo com scripts

- Vá para o nosso módulo em *Examples / Center of Mass*
- Adicionar algumas marcações
- Pressione *Apply*
  - 1. Exibe o centro de deslocamento da posição de massa → funciona!
  - 2. Não exibe nada → erro pode ser visto no interator Python



# Adicionar atualização automática

- Adicionar uma nova etiqueta e caixa de seleção:
  - Altere o nome do objeto para: autoUpdateCheckBox
  - Alterar o texto do rótulo para “**Enable auto-update:**”



- Adicione isso na função de configuração à parte inferior da seção de conexões:

```
self.observedMarkupNode = None
self.markupsObserverTag = None
self.ui.autoUpdateCheckBox.connect("toggled(bool)", self.onEnableAutoUpdate)
```

- Responder a eventos de modificação: adicione-os acima de onApplyButton()

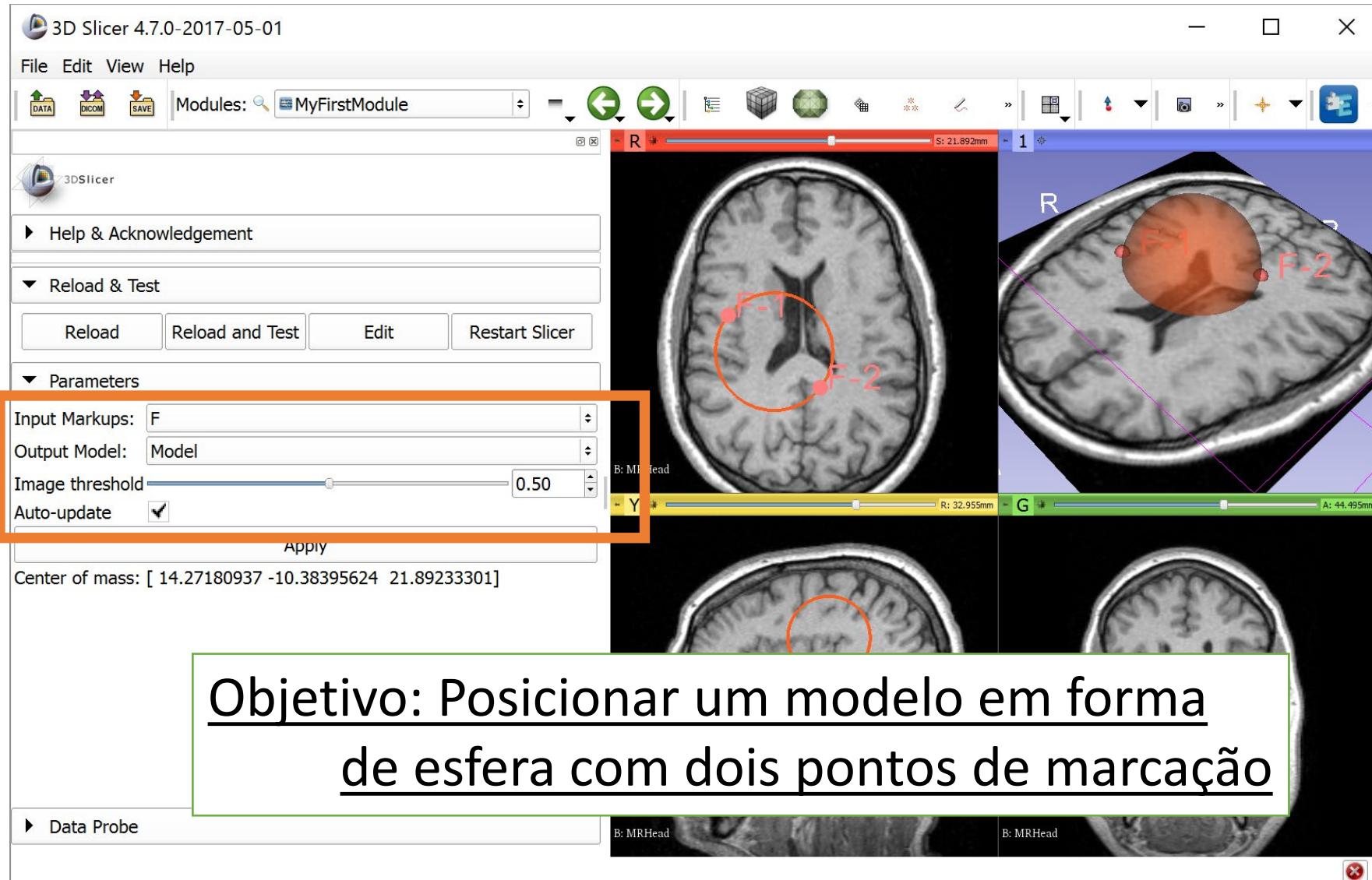
```
def onEnableAutoUpdate(self, autoUpdate):
 if self.markupsObserverTag:
 self.observedMarkupNode.RemoveObserver(self.markupsObserverTag)
 self.observedMarkupNode = None
 self.markupsObserverTag = None
 if autoUpdate and self.ui.inputSelector.currentNode():
 self.observedMarkupNode = self.ui.inputSelector.currentNode()
 self.markupsObserverTag = self.observedMarkupNode.AddObserver(
 slicer.vtkMRMLMarkupsNode.PointModifiedEvent, self.onMarkupsUpdated)

def onMarkupsUpdated(self, caller=None, event=None):
 self.onApplyButton()
```

# Parte 3

Escreva um módulo com script  
(um pouco) simples de forma  
independente

# Descrição da tarefa



# Documentação da API

## Computação genérica e bibliotecas GUI

| Toolkit                                                                       | API documentation                                                                                                         |
|-------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| <a href="https://docs.python.org/es/3.9/index.html">Python</a>                | <a href="https://docs.python.org/es/3.9/index.html">https://docs.python.org/es/3.9/index.html</a>                         |
| <a href="http://docs.scipy.org/doc/numpy/reference">Numpy</a>                 | <a href="http://docs.scipy.org/doc/numpy/reference">http://docs.scipy.org/doc/numpy/reference</a>                         |
| <a href="https://vtk.org/doc/nightly/html">VTK</a>                            | <a href="https://vtk.org/doc/nightly/html">https://vtk.org/doc/nightly/html</a>                                           |
| <a href="http://www.itk.org/SimpleITKDoxygen/html/classes.html">SimpleITK</a> | <a href="http://www.itk.org/SimpleITKDoxygen/html/classes.html">http://www.itk.org/SimpleITKDoxygen/html/classes.html</a> |
| <a href="https://doc.qt.io/qt-5.15/classes.html">Qt</a>                       | <a href="https://doc.qt.io/qt-5.15/classes.html">https://doc.qt.io/qt-5.15/classes.html</a>                               |

## Bibliotecas específicas da segmentação de dados

| Toolkit                                                               | API documentation                                                                                                                                                                                                                                                                                                                              |
|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="http://www.slicer.org/doc/html/classes.html">Slicer core</a> | C++: <a href="http://www.slicer.org/doc/html/classes.html">http://www.slicer.org/doc/html/classes.html</a><br>Python: <a href="http://mwohlke-kitware.github.io/Slicer/Base/slicer.html">http://mwohlke-kitware.github.io/Slicer/Base/slicer.html</a><br>For up-to-date docs, type this into Python console:<br><code>help(slicer.util)</code> |
| <a href="http://www.commonatk.org/docs/html/classes.html">CTK</a>     | <a href="http://www.commonatk.org/docs/html/classes.html">http://www.commonatk.org/docs/html/classes.html</a>                                                                                                                                                                                                                                  |

# Onde encontrar exemplos

- Núcleo do Slicer: <https://github.com/Slicer/Slicer>
- Extensões:
  - Índice de extensões (consulte o repositório em \*.s4ext):  
<https://github.com/Slicer/ExtensionsIndex>
  - SlicerIGT: <https://github.com/SlicerIGT/SlicerIGT/>
  - SlicerRT:  
<https://github.com/SlicerRt/SlicerRT>

# Dicas

- Para poder mostrar um nó do modelo, crie um nó de exibição:  
`outputModel.CreateDefaultDisplayNodes()`
- Para mostrar interseções de modelo com um visualizador de fatia 2D:  
`outputModel.GetDisplayNode().SetSliceIntersectionVisibility(True)`
- Lembre-se de que toda a lógica é implementada na função de execução na classe lógica, que é chamada na função `onApplyButton`

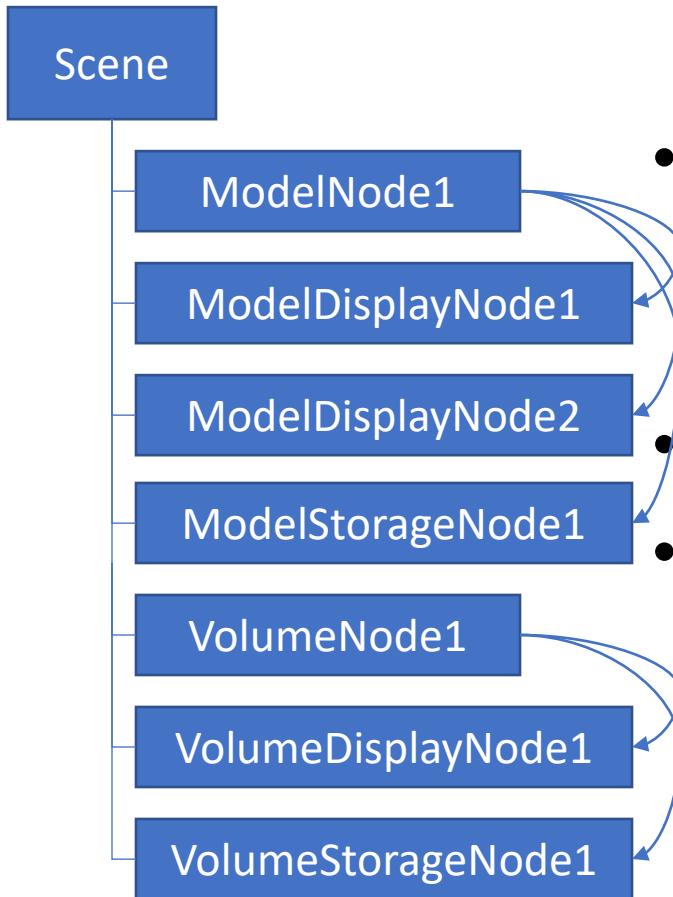
# Apêndice

# Transformações

- Sistema de coordenadas mundial de segmentação de dados: RAS (direita-ântero-superior)
- Obter transformação linear do nó para RAS:

```
nodeToRas = vtk.vtkMatrix4x4()
if node.GetTransformNodeID():
 nodeToRasNode = slicer.mrmlScene.GetNodeByID(node.GetTransformNodeID())
 nodeToRasNode.GetMatrixTransformToWorld(nodeToRas)
```
- A transformação pode ser não-linear
- Pelo menos registre um erro se a transformação estiver presente, mas ela for ignorada
-

# Referências de nós



- Use isso sempre que um nó depender de dados armazenados em outros nós
  - Especificado por nome de papel, ID do nó referenciado, índice (várias referências com a mesma função são permitidas)
- Salvo/restaurado com a cena**
- Não é trivial: ao importar uma cena e uma ID de nó já é encontrada na cena atual, a ID de nó importada é renomeada automaticamente e todas as referências são atualizadas

# Design da GUI

- O Qt designer pode ser usado
- O arquivo de interface do usuário gerado pode ser carregado para criar a GUI do módulo:

<http://www.slicer.org/slicerWiki/index.php/Documentation/Nightly/Developers/Tutorials/PythonAndUIFile>

# Visão geral das APIs acessíveis a partir do Python



NumPy

