Images

10.14.24

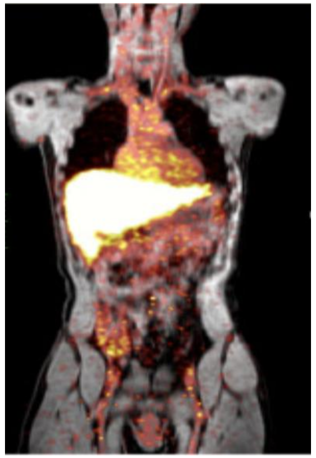
Learning Objectives

- Preprocessing
- Segmentation
- Transformations

What types of info can an “image” contain?

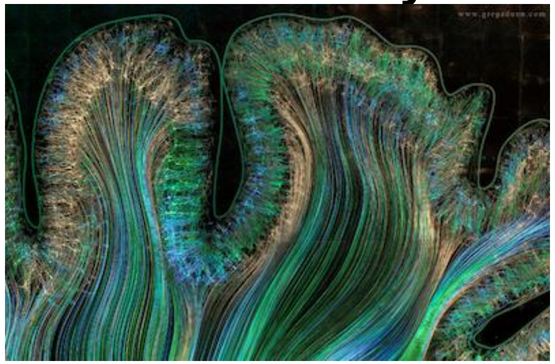
What types of info can an “image” contain?

Composition



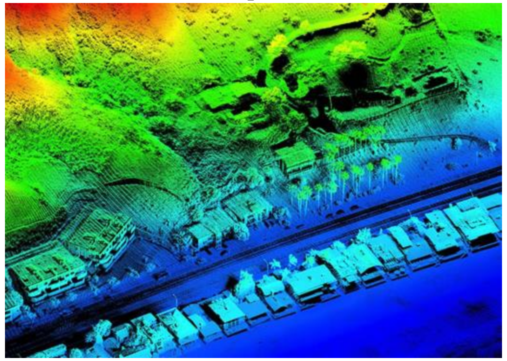
PET

Geometry



Diffusion-Tensor Image

Depth

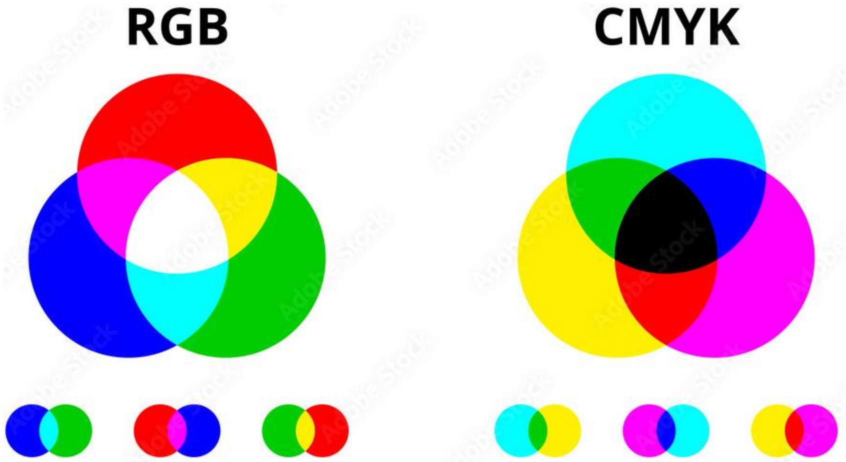


LIDAR

Temperature

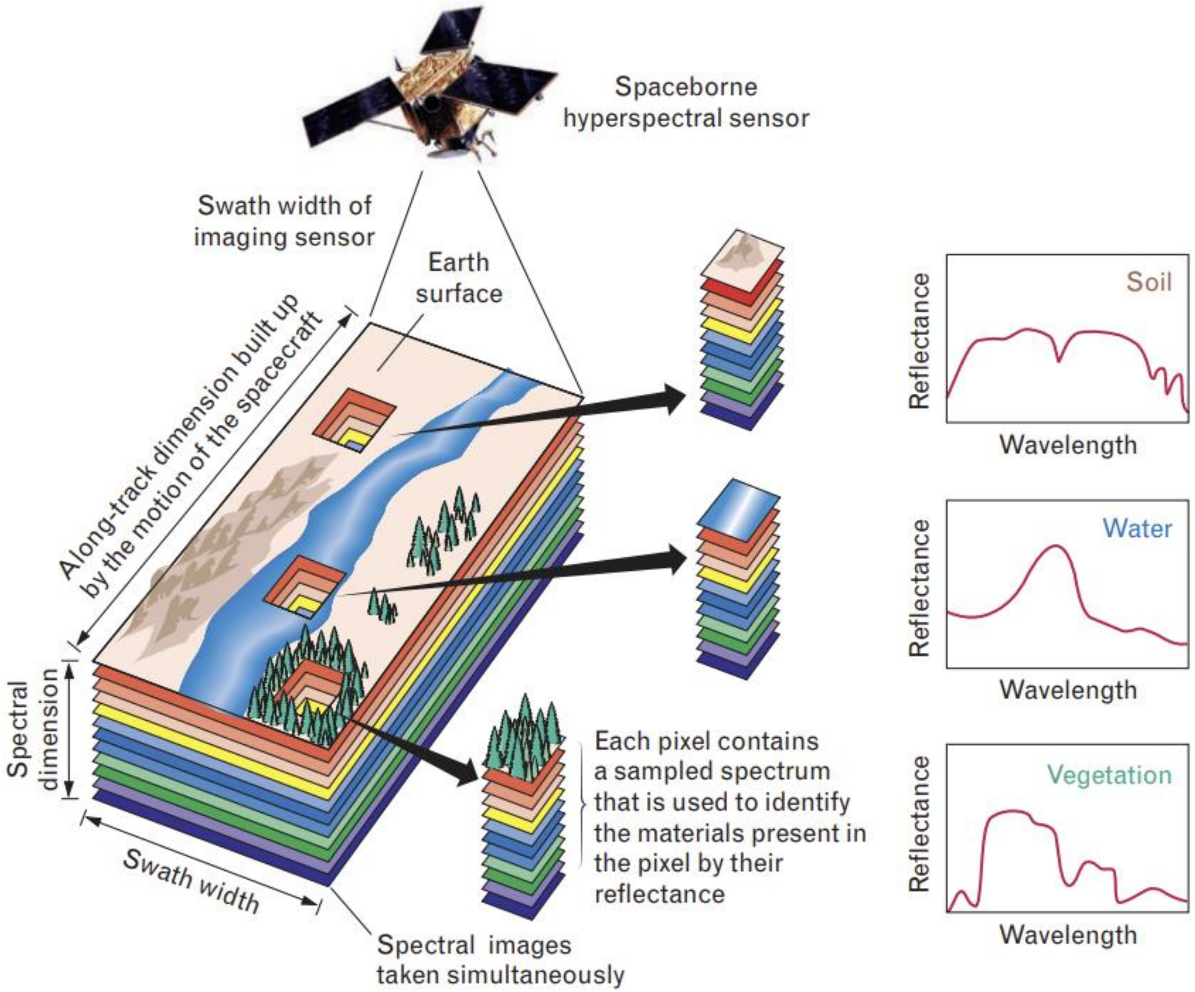
Infrared

“Color” in camera-imaging



“Color” in camera-imaging

Hyperspectral Imaging

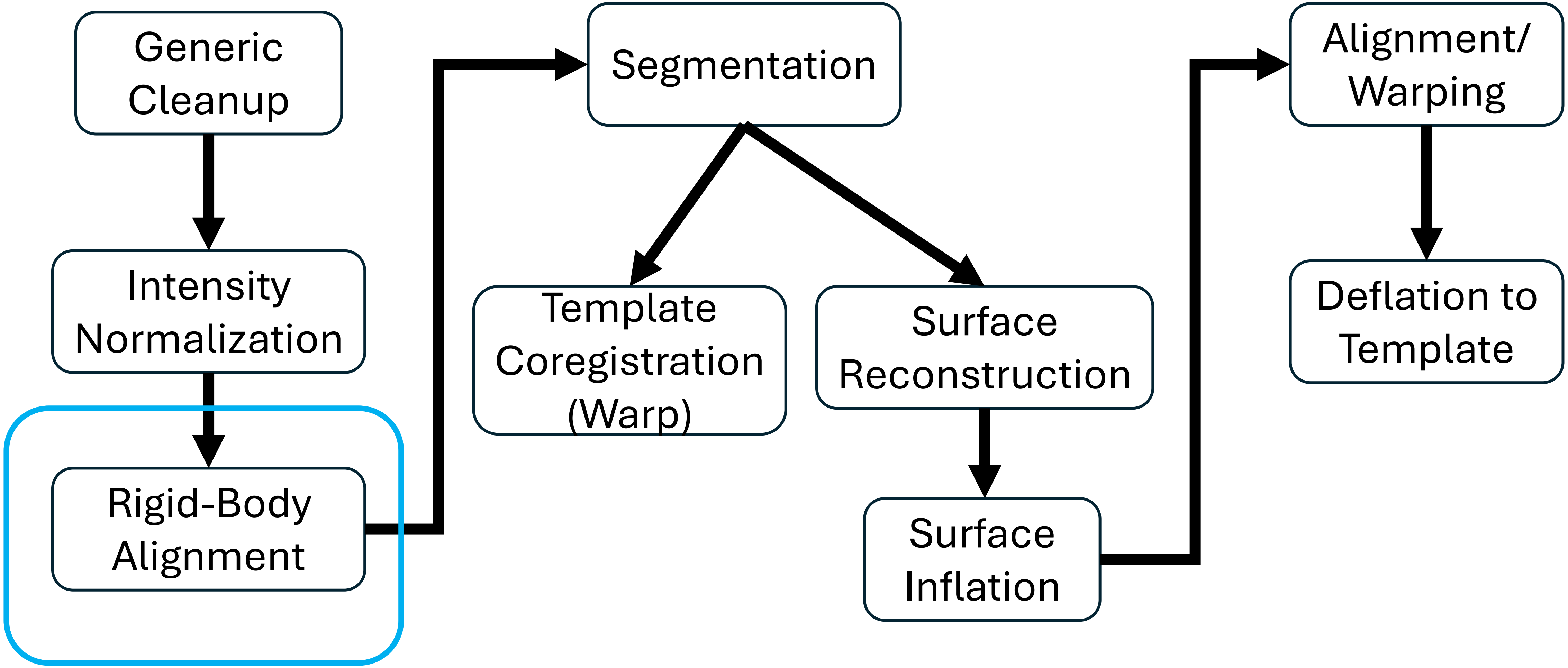


What are some challenges with image-data?

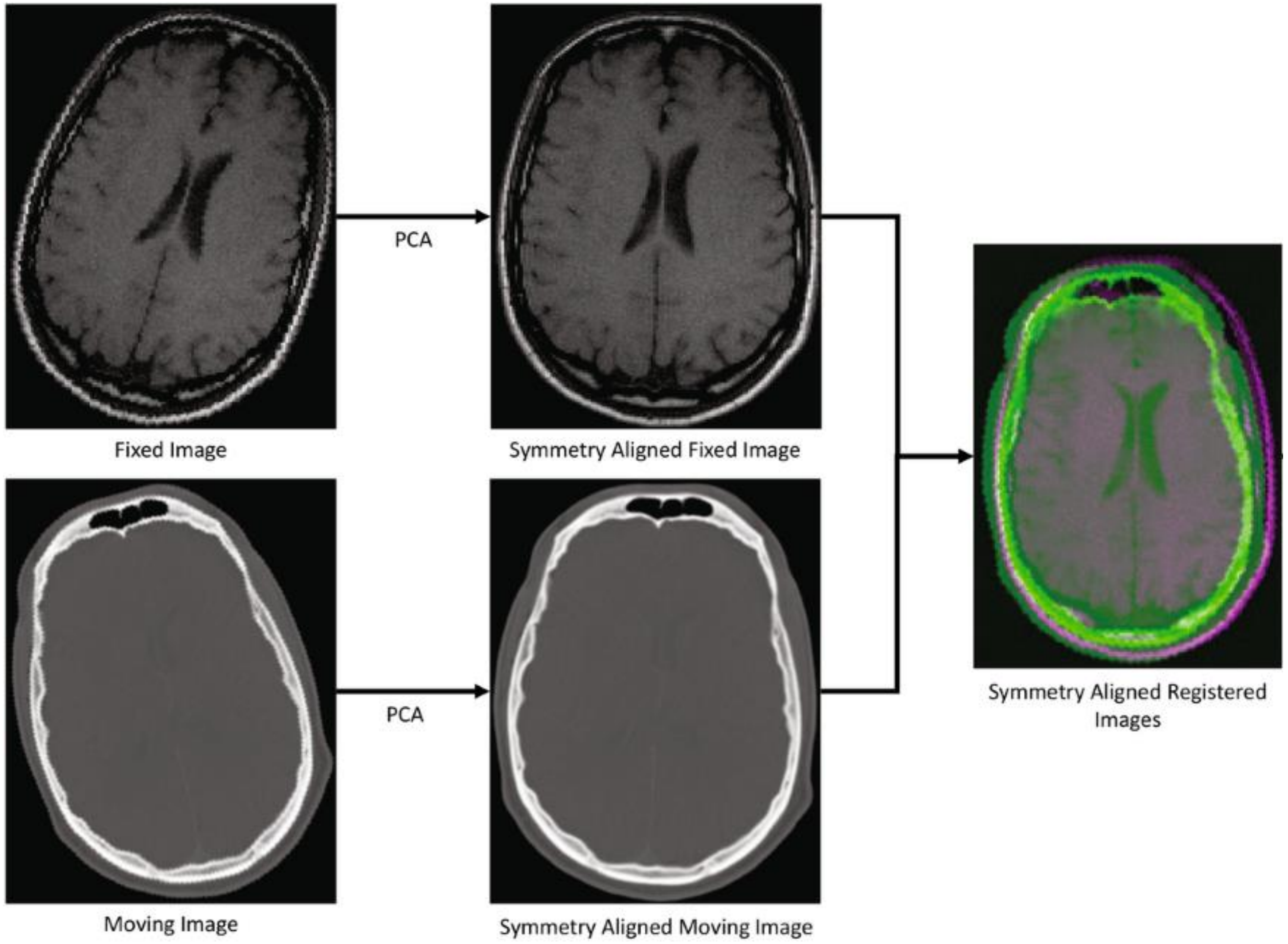



Some example images for “computer”

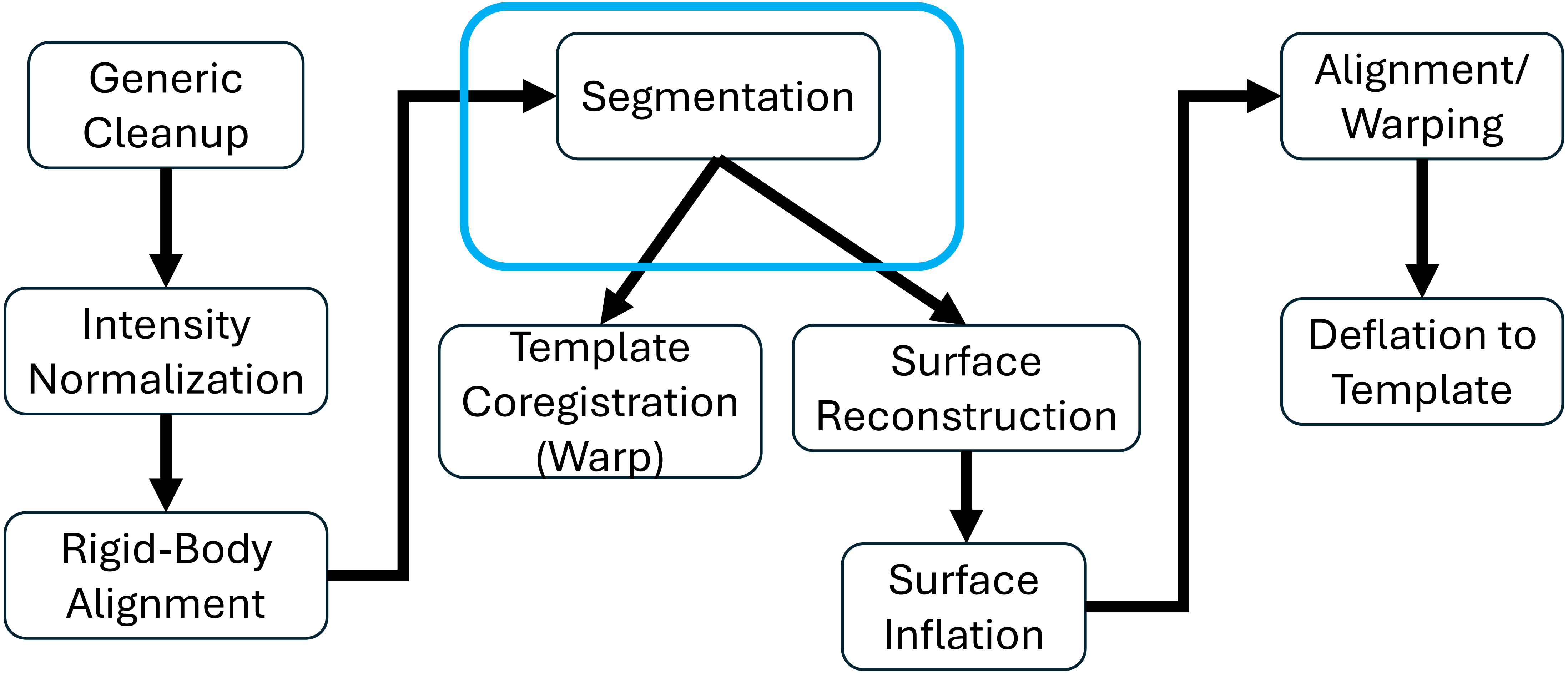
A typical fMRI processing pipeline



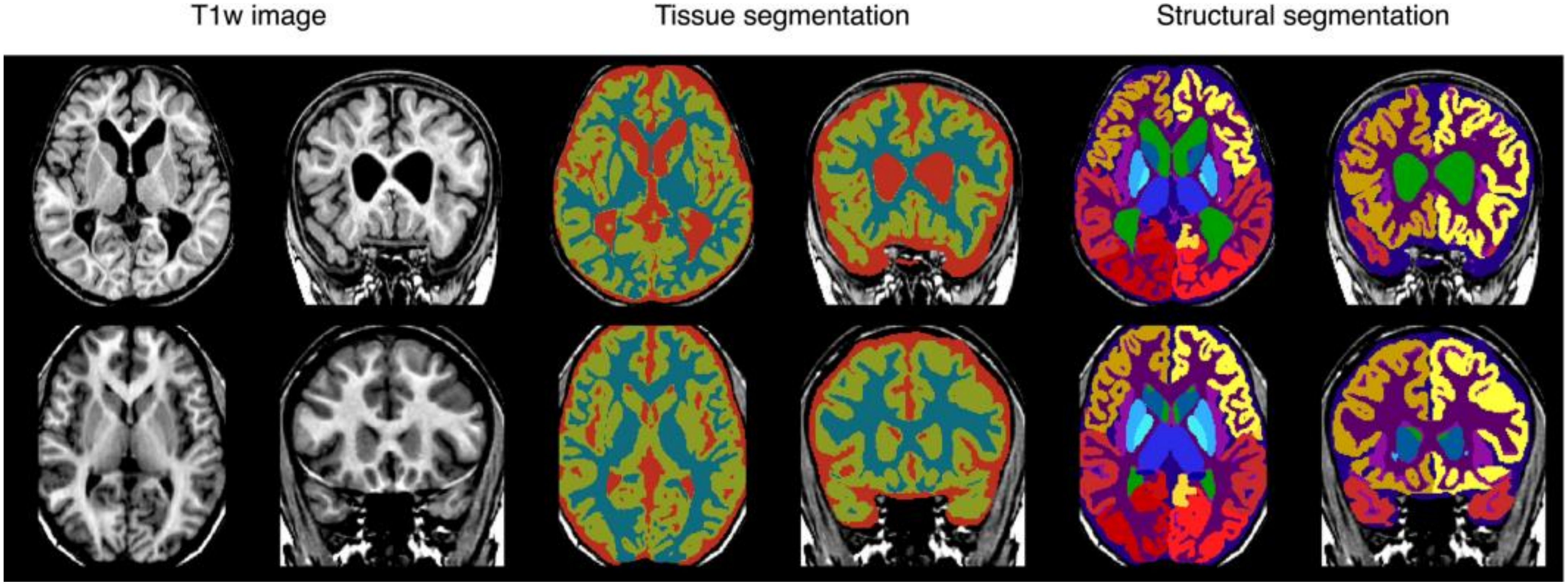
Registration



A typical fMRI processing pipeline



Segmentation



Imaging Modules in Python

- opencv
  - Much faster. Designed for real-time application
  - C++ and builtin CUDA integration (GPU)
  - Color is BGR

- **scikit-image**
  - Slower
  - Easier access to basic processing functions
  - <u>Built-in generalization to ND images (e.g., volumes)</u>
  - <u>Color is RGB</u>

Scikit-Image Submodules

- Data: classic examples in image-processing

astronaut

coffee

camera

coins

```
coins = skimage.data.coins()
```

Scikit-Image Submodules

- **Feature**
- **Filters**
- **IO**
- **Morphology**
- **Restoration**
- **Segmentation**
- **Transform**

Processing: Filtering

Have I ever mentioned convolution?

Image Restoration

- Smoothing: Gaussian, box (movavg), median

```
from skimage import data,io,filters
from matplotlib import pyplot as plt
import numpy as np
from numpy import random as random

cam=data.camera();
## Poisson noise
camNoiseE=cam+random.exponential(10,np.shape(cam));
## Uniform noise
camNoiseU=cam+random.random(np.shape(cam))*20;
```

Jupyter

Image Restoration

- Convolution model of blurring

$$X_{meas} = X_{true} * M_{blur} + \eta_{noise}$$

- Convolution Theorem:

$$\mathcal{F}[A * B] = \mathcal{F}[A] \cdot \mathcal{F}[B]$$

$$X_{true} \approx \mathcal{F}^{-1}\left(\frac{\mathcal{F}[X_{meas}]}{\mathcal{F}[M_{blur}]}\right)$$

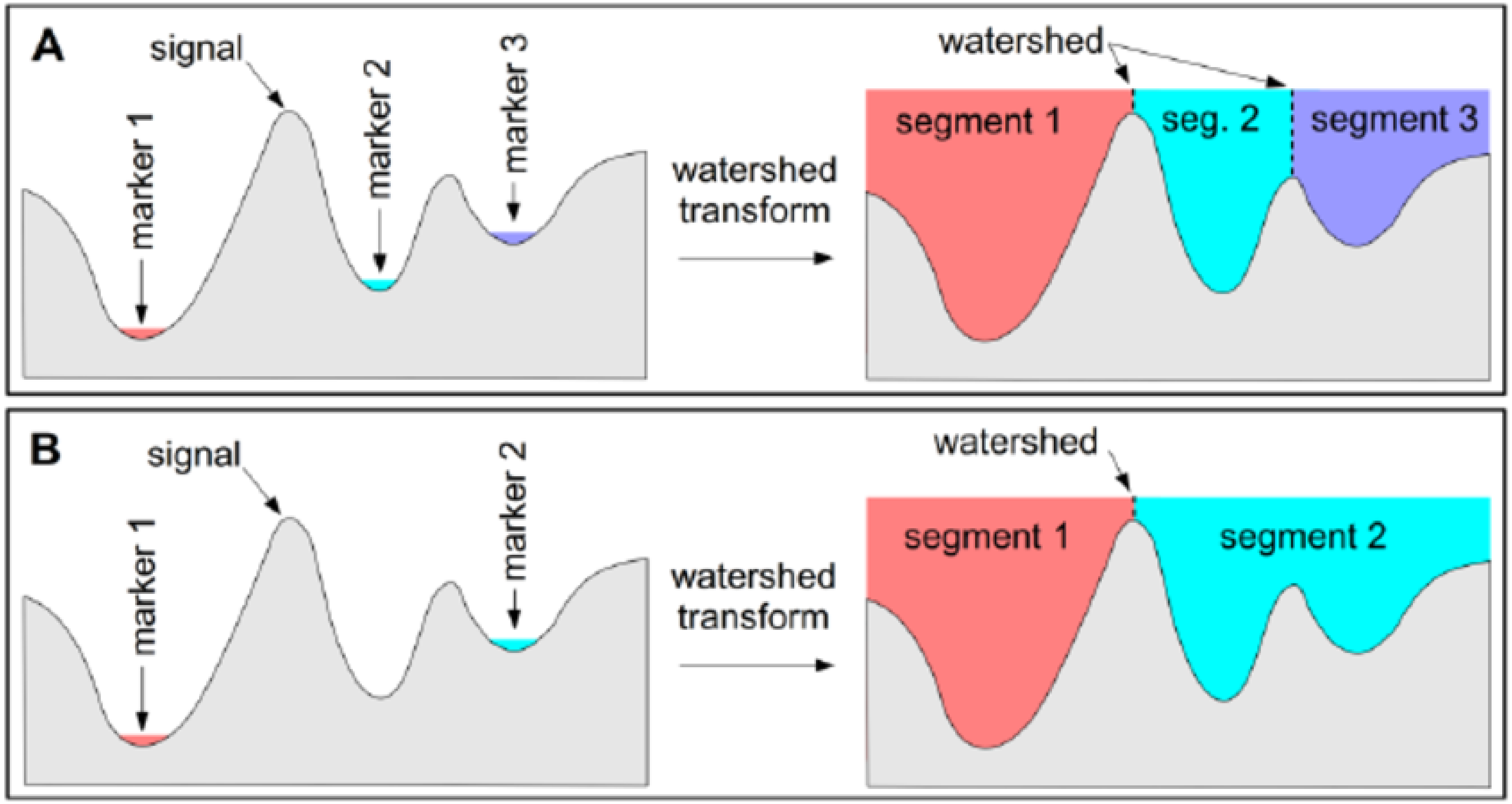
Gaussian η: Wiener Filter

Poisson η: Richardson-Lucy

Image Restoration

Input Image | Simulate blur and Noise | Restored Image

Segmentation: Watershed Algorithm



Sobel Gradient Approximation

- Edges are where the change in intensity (gradient) is large
- Sobel kernels: gradient in one direction, smoother in the other:

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \qquad G_y = G_x^T$$

- Full Sobel is gradient magnitude: $G_x^2 + G_y^2$

Coins Example

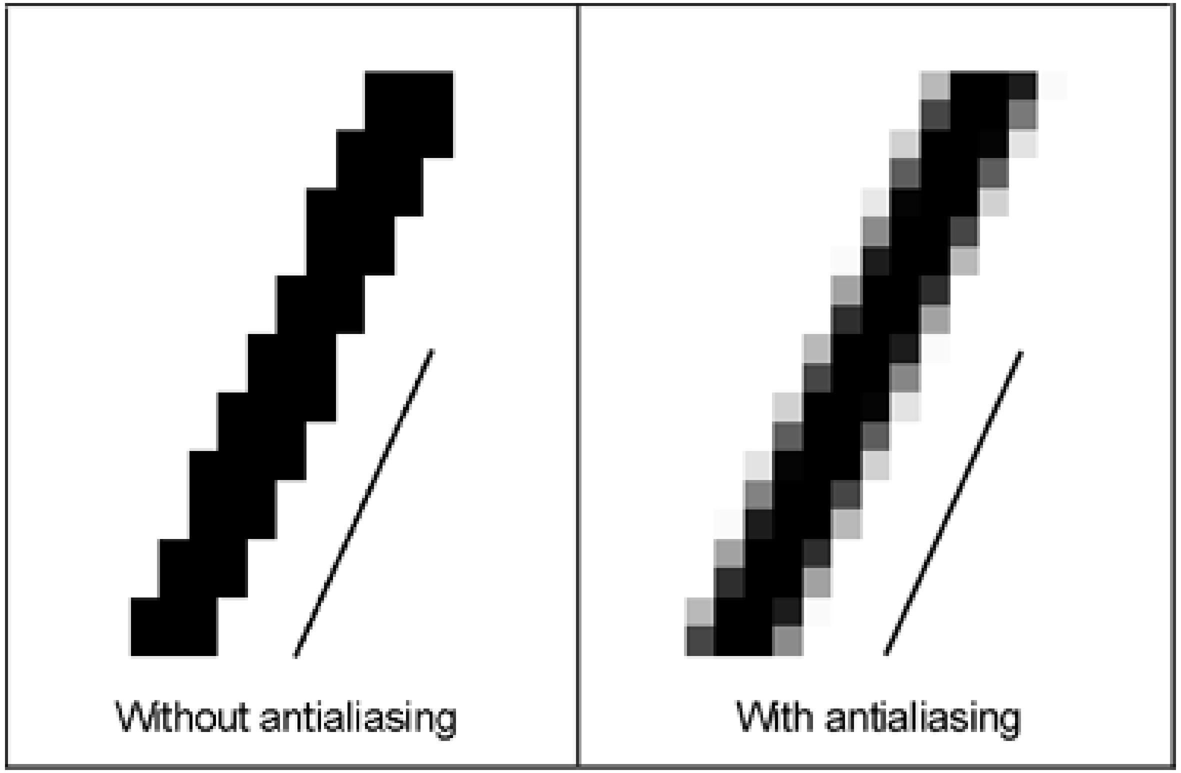
- Jupyter

Practice: Edge detection

- Find values to separate the image into coins and background
- Apply the Sobel filter: filters.sobel and create an image of gradient intensity

Resizing

- rescale: Isotropic (same in each direction)
- resize: arbitrary dimensions
- antialiasing: usually Gaussian smoothing kernel



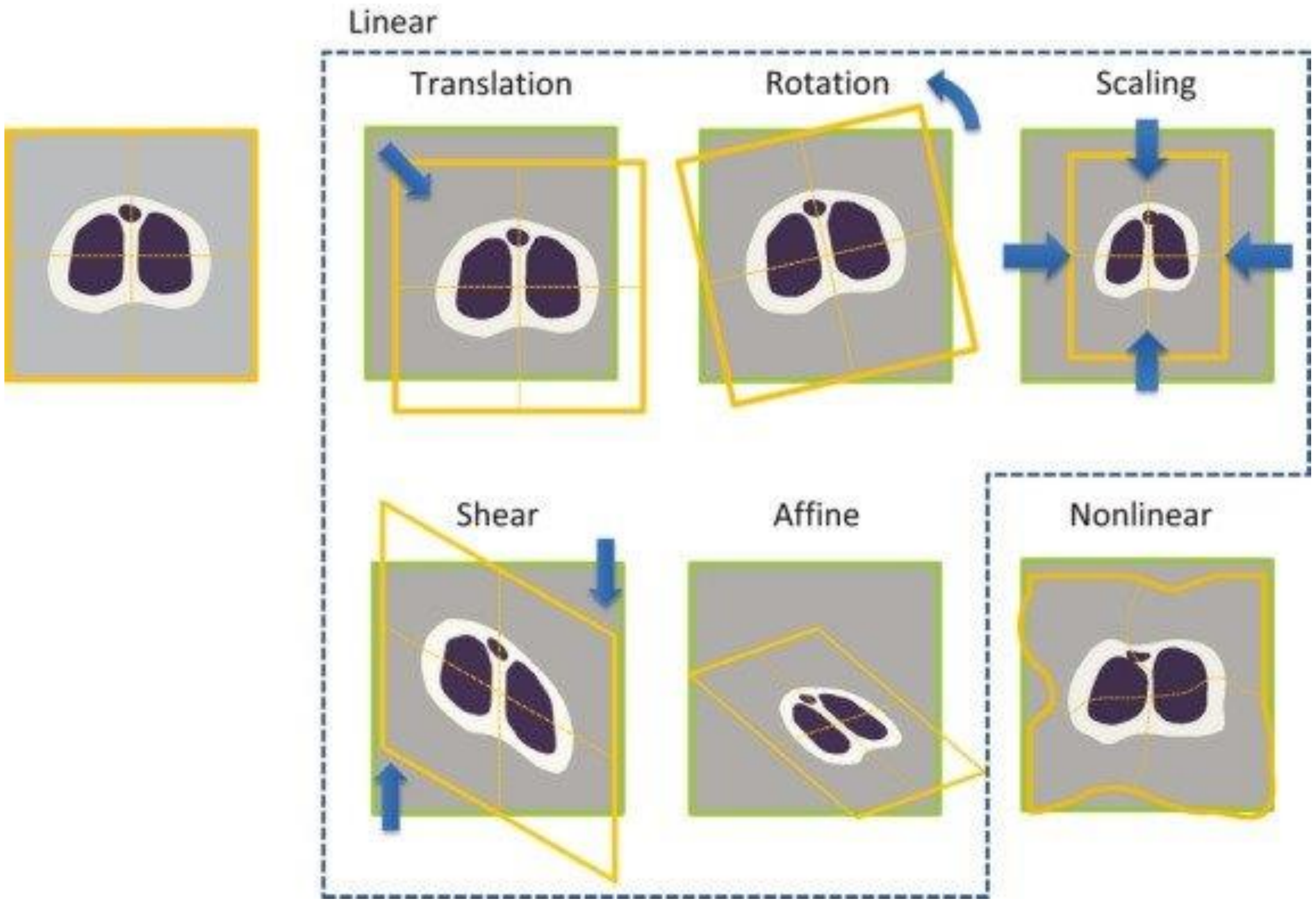
Transformation-Types

- Euclidean: Size-Preserving (rigid-body)
  - Translation, rotation

- Affine
  - Translation + Linear operator
  - Includes resizing and shearing transforms

- Nonlinear
  - Curvilinear warping
  - Custom coordinate-systems

Transformation Types



Learning Objectives

- Preprocessing
- Segmentation
- Transformations

Fin