PRESENTER:
Konstantinos Ntatsis

itk-elastix: Medical image registration in Python *

BACKGROUND: itk-elastix is a pythonic wrapping of the scientific image registration toolbox *elastix*.

Elastix
- **... is based on ITK**
- **... offers modular design**
- **... has two decades of development**

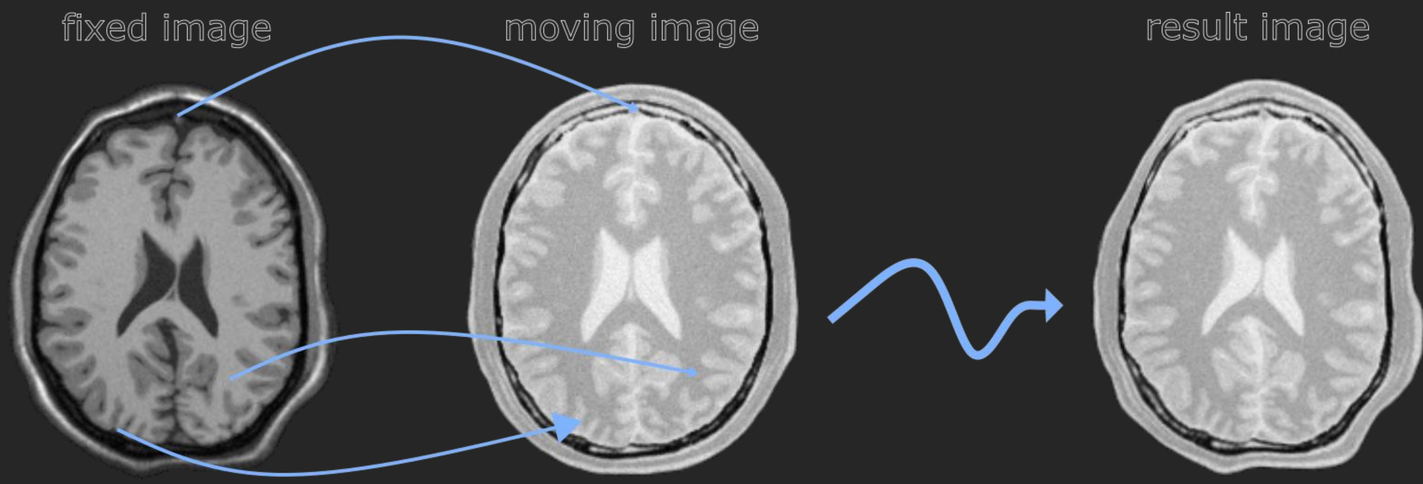
Key characteristics
- **Registration of 2D, 3D, 4D images**
- **Rigid / Affine / Non-rigid (B-splines)**
- **Multi-metric, Multi-resolution**
- **Setting masks/points to aid registration**
- **Transform images, meshes and point sets**

Integration with project MONAI
Combine itk-elastix with the pytorch-based medical deep learning library MONAI

Jupyter Notebook tutorials
To get started quickly, see /examples

Model zoo
Collection of published parameter files at https://elastix.lumc.nl/modelzoo/

Integration with Napari as plugin
https://github.com/SuperElastix/elastix-napari



GitHub repo

*** *pip install itk-elastix* today to use it for free**

https://github.com/InsightSoftwareConsortium/ITKElastix

Co-authors:

Konstantinos Ntatsis, Niels Dekker, Viktor van der Valk, Tom Birdsong, Dženan Zukić, Stefan Klein, Marius Staring, Matthew McCormick

Leids Universitair Medisch Centrum

Erasmus MC

kitware

CODE SNIPPET: Registration

```
import itk
from scipy.spatial.distance import dice

# Load the moving and the fixed image from disk
fixed_image = itk.imread('./data/fixed.mha', itk.F)
moving_image = itk.imread('./data/moving.mha',
itk.F)

# Configure a (default) parameter map with all the
# registration parameters
par_obj = itk.ParameterObject.New()
par_map = par_obj.GetDefaultParameterMap('bspline')
par_obj.AddParameterMap(par_map)

# Run the registration
# rtp: result transform parameter object
result_image, rtp = itk.elastix_registration_method(
    fixed_image, moving_image,
    parameter_object=par_obj)
```

CODE SNIPPET: Mask transformation & Dice calculation

```
(...continuing from the previous snippet)

# Load the corresponding masks
fixed_mask = itk.imread('./data/f_mask.mha', itk.UC)
moving_mask = itk.imread('./data/m_mask.mha', itk.UC)

# Transform the moving mask using the result from the
# registration
rtp.SetParameter(0,'ResampleInterpolator',
                 'FinalNearestNeighborInterpolator')
result_mask = itk.transformix_filter(moving_mask, rtp)

# Compute dice on masks
initial_dice = 1 - dice(fixed_mask[:].ravel(),
                        moving_mask[:].ravel())
result_dice = 1 - dice(fixed_mask[:].ravel(),
                       result_mask[:].ravel())
print(initial_dice, result_dice)
```

Parameter map/file example

```
(AutomaticParameterEstimation "true")
(CheckNumberOfSamples "true")
(DefaultPixelValue 0)
(FinalBSplineInterpolationOrder 3)
(FinalGridSpacingInPhysicalUnits 10)
(FixedImagePyramid "FixedGenericImagePyramid")
(GridSpacingSchedule 2.80322 1.9881 1.41 1)
(ImageSampler "RandomCoordinate")
(Interpolator "LinearInterpolator")
(MaximumNumberOfIterations 256)
(MaximumNumberOfSamplingAttempts 8)
(Metric "AdvancedMattesMutualInformation")
(MovingImagePyramid
"MovingGenericImagePyramid")
(NewSamplesEveryIteration "true")
(NumberOfResolutions 4)
(NumberOfSamplesForExactGradient 4096)
(NumberOfSpatialSamples 2048)
(Optimizer
"AdaptiveStochasticGradientDescent")
(Registration "MultiResolutionRegistration")
(Transform "BSplineTransform")
```