TP Insight Toolkit

roman.fenioux@kitware.com

Le but de ce TP est de se familiariser avec la bibliothèque ITK (Insight Toolkit) et d'utiliser les méthodes de recalage présentes dans cette bibliothèque. ITK (http://www.itk.org) est une bibliothèque de traitement d'images qui n'a aucune interface graphique. Nous produirons donc des images 2D qui seront visualisées avec un logiciel de visualisation d'images générique (Gimp ou autres). Il est conseillé de se référer à la documentation d'ITK (compatible avec la version installée sur les machines) :

https://www.itk.org/Doxygen50/html/classes.html

Exercice 1 : ITK Hello World

Le but de cet exercice est d'écrire un script Python qui affiche la version d'ITK installée. Cette application permet juste de vérifier qu'ITK est bien installé et fonctionne correctement.

Astuces :

1. On importera le module python ***itk***
2. Il faudra chercher dans la documentation (ou sur le web) comment afficher la version du package.

Exercice 2 : Lecture d'une image

Le but de cet exercice est de lire l'image « itklogo.jpg » et d'afficher la valeur du pixel à l'index (10,10) sur la console.

Astuces :

1. On utilisera la classe « ImageFileReader » pour cela
2. Cette classe est « templatée » sur le type d'image. Nous pourrons utiliser un type unsigned char.

Questions :

1. Quelle est la valeur du pixel à l'index [10,10] ?
2. Quelle est la valeur du pixel à l'index [600,133] ?
3. Peut-on utiliser un type d'image RGB ?
4. Il y'a-t-il une méthode plus simple pour lire une image ?

Exercice 3 : Lecture et écriture d'une image

Modifier le code de l'exercice précédent afin d'écrire l'image qui vient d'être lue sur le disque dur sous le nom « imagesortie.jpg » et « imagesortie.bmp ».

Astuces :

1. On utilisera la classe « ImageFileWriter » pour cela

Questions :

1. Comment peut on essayer de simplifier le code au minimum ?

Exercice 4 : Filtrage d'image

En utilisant le filtre itkRecursiveGaussianImageFilter, écrire un programme qui lit « itklogo.jpg » et applique un filtre Gaussien sur l'image.

Astuces :

1. Les filtres d'ITK sont templatés. Il faut trouver les bonnes valeurs de template.
2. Attention certains types de pixel ne sont pas supporté par le wrapping Python d'ITK

Questions :

1. Quelle est le paramètre pour modifier la valeur du flou ?
2. Comment peut-on appliquer le filtre dans toutes les directions ?
3. Détecter les "edges" dans l'image avec le filtre itkRecursiveGaussianImageFilter
4. Si vous avez le temps vous pouvez modifier votre code pour utiliser le filtre *itkAbsoluteValueDifferenceImageFilter* afin de calculer la différence entre deux images.

Exercice 5 : Segmentation d'image

Nous allons segmenter l'image « brain.png » en utilisant le filtre itkConnectedThresholdImageFilter

Astuces :

1. Vous aurez de meilleurs résultats en appliquant un filtre itkGradientAnisotropicDiffusionImageFilter en amont du pipeline de traitement.
2. Vous pouvez utiliser l'index [110,100] comme point de départ
3. Pour écrire l'image finale vous aurez surement besoin d'utiliser le filtre itkRescaleIntensityImageFilter.

Exercice 6 : Recalage par Translation

Ecrire un programme qui effectue le recalage de l'image **BrainProtonDensitySliceShifted13x17y.png** (moving) avec **BrainProtonDensitySliceBorder20.png** (fixed) en utilisant :

- Transformation : itkTranslationTransform
- Métrique : itkMeanSquaresImageToImageMetricv4
- Interpolateur : itkLinearInterpolateImageFunction
- Optimiseur: itkRegularStepGradientDescentOptimizerv4

La classe qui lie tous ces éléments est itkImageRegistrationMethodv4

Astuces :

1. Pour lancer l'optimisation il faut appeler Update() sur itkImageRegistrationMethodv4
2. Récupérez les paramètres finaux de la transformation après optimization
3. Pour afficher le résultat du recalage on utilisera la classe : itkResampleImageFilter.

Questions :

1. Essayer de diminuer le nombre d'itérations pour voir comment l'optimisation se comporte

Exercice 7 : Recalage par transformation rigide

A partir de l'exercice précédent, écrire un programme qui effectue le recalage de l'image **BrainProtonDensitySliceR10X13Y17.png** (moving) avec **BrainProtonDensitySliceBorder20.png** (fixed) en utilisant une transformation rigide.

Astuces :

1. Il peut être utile de fixer le centre de rotation.
2. Il peut être nécessaire d'utiliser la fonction SetScales() sur l'optimizer

Exercice 8 : Recalage par information mutuelle

A partir de l'exercice précédent, écrire un programme qui effectue le recalage de l'image **BrainProtonDensitySliceR10X13Y17.png** (moving) avec **BrainT1SliceBorder20.png** (fixed) en utilisant une transformation itkSimilarity2DTransform et la métrique d'information mutuelle par Mattes (itkMattesMutualInformationImageToImageMetricv4).

Questions :

1. Quels sont les paramètres de cette métrique qui peuvent influencer le résultat du recalage ?