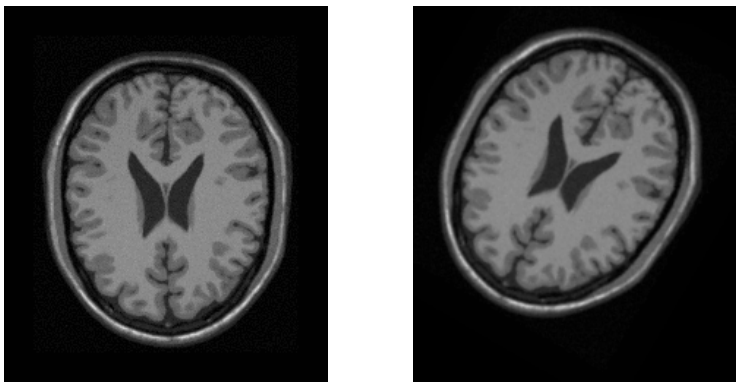
Tutorial: Image Registration

Note

This tutorial aims to implement the Iterative Closest Point (ICP) method for image registration. More specifically, we are going to estimate a rigid transformation (translation + rotation without scaling) between two images.

The different processes will be applied on T1-MR images of the brain in Fig.1.



(a) brain1 (b) brain2

Figure 1: Original images.

1 Transformation estimation based on corresponding points

A classical method in image registration first consists in identifying and matching some characteristic points by pairs. Thereafter, the transformation is estimated from this list of pairs (displacement vectors).

1.1 Preliminaries

Pairs of points are first manually selected.

- Read and visualize the two MR images 'brain1' and 'brain2' (moving and source images).

- Manually select a list of corresponding points.

Informations

cpselect is a graphical interface dedicated to manual selection of pairs of points.

Informations

There is no built-in function for manual selection of pairs of points. You can use the following points or code your own function.

```
# define control points
A_points = np.array([[136, 100], [127, 153], [96, 156], [87, 99]]);
B_points = np.array([[144, 99] , [109, 140], [79, 128], [100, 74]]);
```

1.2 Rigid transformation

With this list of pairs $(p_i, q_i)_i$, this tutorial proposes to estimate a rigid transformation between these points. It is composed of a rotation and a translation (and not scaling). For doing that, we make a Least Squares (LS) optimization, which is defined as follows. The parameters of the rotation R and the translation t minimize the following criterion:

$$C(R,t) = \sum_i ||q_i - R.p_i - t||^2$$

Calculation of the translation :
The optimal translation is characterized by a null derivative of the criterion:

$$\frac{\partial C}{\partial t} = -2\sum_i (q_i - R.p_i - t)^{\mathrm{T}} = 0 \Leftrightarrow \sum_i q_i - R.\left(\sum_i p_i\right) = N.t$$

where N denotes the number of matching pairs. By denoting $\bar{p} = \frac{1}{N}\sum_i p_i$ and $\bar{q} = \frac{1}{N}\sum_i q_i$ the barycenters of the point sets and by changing the geometrical referential: $p'_i = p_i - \bar{p}$ et $q'_i = q_i - \bar{q}$, the criterion can be written as:

$$C'(R) = \sum_i ||q'_i - R.p'_i||^2$$

The estimated rotation $\hat{R}$ will provide the expected translation:

$$\hat{t} = \bar{q} - \hat{R}.\bar{p} \tag{1}$$

calculation of the rotation by the SVD method:
We will use the following theorem: Let $U.D.V^{\mathrm{T}} = K$ a singular decomposition of the correlation matrix $K = {q'}^{\mathrm{T}}.p'$, for which the singular values are sorted in the increasing order. The minimum of the criterion: $C(R) = \sum_i ||q'_i - R.p'_i||^2$ is reached by the matrix :

$$\hat{R} = U.S.V^{\mathrm{T}} \tag{2}$$

with $S = DIAG(1, \ldots, 1, \det(U).\det(V))$.

- Code a function that takes as parameters the pairs of points, and returns the rigid transformation elements of rotation $\hat{R}$ and translation $\hat{t}$ as defined in Eqs.1 and 2.
- Apply the transformation to the moving image.
- Visualize the resulting registered image.

Informations

- See svd function.
- See imwarp and affine2d for transformation construction and application.
- See imshowpair for displaying the registration results.

Informations

- See svd function from scipy.linalg or numpy.linalg.
- See cv2.warpAffine and cv2.transform for transformation application

2 ICP-based registration

When the points are not correctly paired, it is first necessary to reorder them before estimating the transformation. In this way, the ICP (iterative Closest Points) consists in an iterative process of three steps: finding the correspondence between points, estimating the transformation and applying it. The process should converge to the well registered image.

To simulate the mixing of the points, randomly shuffle them selected on the first image and perform the registration with the previous method.

Informations

See randperm.

Informations

See np.random.permutation.

1. From the list of the characteristic points p of the image 'brain1', find the nearest neighbors q in the image 'brain2'. Be careful to the order of the input arguments.
2. Estimate the transformation T by using the LS minimization coded previously.
3. Apply this transformation to the points p, find again the correspondence between these resulting points $T(p)$ and q and estimate a new transformation. Repeat the process until convergence.
4. Visualize the resulting registered image.

Informations

Look at dsearchn for nearest neighbor search.

Informations

Look at scipy.spatial.cKDTree for nearest neighbor search.

3 Automatic control points detection

The manual selection of points may be fastidious. Two simple automatic methods for detecting control points are the so-called Harris corners detection or Shi-Tomasi corner detector.

Informations

The function detectHarrisFeatures returns the corners, their coordinates can be retrieved by corners.selectStrongest(nb_points).

Informations

The function goodFeaturesToTrack returns the corners from the Shi-Tomasi method.

Replace the manual selection in the two previous parts of this tutorial.

Notice that the automatic points detection does not ensure to give the same order in the points of the two images. More generally, other salient points detectors do not give the same number of points, and thus the algorithms have to remove outliers (non matching points). This case is not taken into account in this tutorial.