Week 9: Landmark Image Registration

Instructor: Daniel L. Pimentel-Alarcón

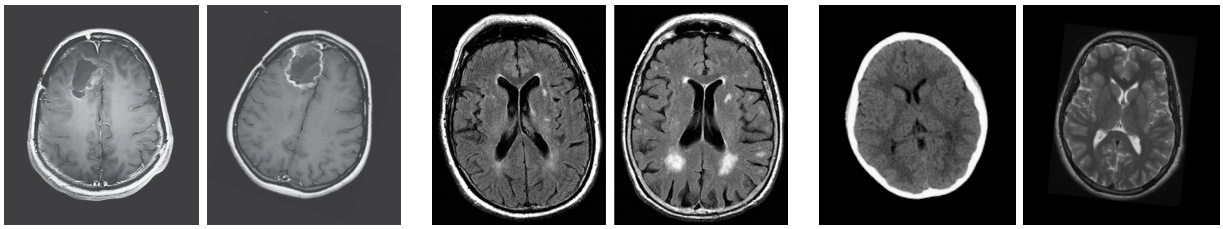
GO GREEN. AVOID PRINTING, OR PRINT 2-SIDED MULTI-PAGE.

9.1 Introduction

Medical images are rarely perfectly aligned. For example, because patients move, have different positions in different sessions, or simply because they are different patients and have different dimensions and proportions. However, in many cases we need images to be aligned, for example:

- To measure a treatment effectiveness.
- To compare healthy vs. unhealthy samples.
- To assess complementary information.

The process of aligning images is often called *image registration*.



(a) (b) (c)

Figure 9.1: (a) Two MRI's of the same patient taken before and after a treatment. We want to compare them to evaluate the treatment effectiveness. (b) Two MRI's of two different patients, one healthy and one with Alzheimers. We want to identify differences to better understand the disease. (c) CT and MRI scans of the same patient on the same day. We want to study their complementary information.

9.2 Review: Spatial Transformations

Recall from previous lectures that we can transform an image (e.g., rotation, translation, scaling) by changing the positions of the pixels in an image. For example, we can rotate an image $\mathbf{X}$ by an angle θ by changing pixels at locations (x, y) to locations (x', y'), where:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Similarly, we can scale an image $\mathbf{X}$ by fractions α_x and α_y by changing pixels at locations (x, y) to locations (x', y'), where

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \alpha_x & 0 \\ 0 & \alpha_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

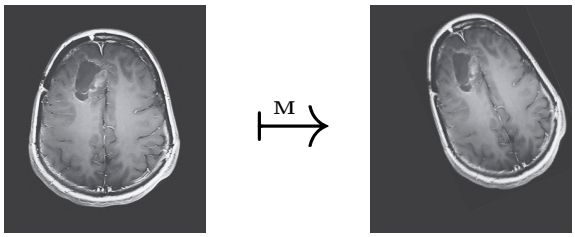
We can also translate an image by mapping (x, y) to $(x + t_x, y + t_y)$, which we can write in matrix form as:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Furthermore, we can combine these transformations in a single matrix operator $\mathbf{M}$ that simultaneously rotates, scales, and translates an image:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Here $\mathbf{X}$ will be transformed depending on the values that $\mathbf{M}$ takes.



$\mathbf{X}$ $\mathbf{X}'$

Figure 9.2: Left: MRI image $\mathbf{X}$. Right: Transformed image $\mathbf{X}'$ (rotated, translated, and scaled) after applying a mapping $\mathbf{M}$ that moves pixels in position (x, y) into new locations (x', y').

9.3 Image Registration

In spatial transformations we know the image $\mathbf{X}$, we know how we want to transform it (i.e., we know the mapping $\mathbf{M}$), and we can easily compute the transformation $\mathbf{X}'$. Image registration is essentially the inverse problem: we know the images $\mathbf{X}$ and $\mathbf{X}'$, and we want to find the mapping $\mathbf{M}$ that aligns $\mathbf{X}$ with $\mathbf{X}'$.

In other words, we want to find the mapping $\mathbf{M}$ that moves the pixels located at (x, y) in $\mathbf{X}$, into locations (x', y'), so that $\mathbf{X}$ and $\mathbf{X}'$ are as aligned as possible. In general, this can be very difficult (specially if the required mapping is non-linear), and there is a lot of active research in this problem.

9.4 Landmark Image Registration

Here we will use a basic yet useful technique, known as *landmark image registration*. The main idea is as follows: suppose we know a collection of coordinates $\{(x_k, y_k)\}_{k=1}^K$ in image $\mathbf{X}$ that correspond to coordinates $\{(x'_k, y'_k)\}_{k=1}^K$ in image $\mathbf{X}'$. That is, for every $k = 1, \ldots, K$, pixel in location (x_k, y_k) in $\mathbf{X}$ corresponds to

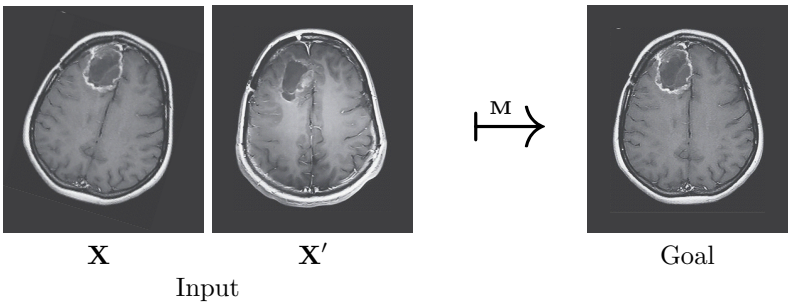


Figure 9.3: Image registration: Given $\mathbf{X}$ and $\mathbf{X}'$, the goal is to find a transformation $\mathbf{M}$ that aligns $\mathbf{X}$ with $\mathbf{X}'$.

pixel in location (x'_k, y'_k) in $\mathbf{X}'$. These points are called *landmarks*, and that is where this method gets its name. The goal is to find a matrix $\mathbf{M}$ that relocates all pixels located at (x_k, y_k) into positions (x'_k, y'_k) via the following mapping:

$$\begin{bmatrix} x'_k \\ y'_k \end{bmatrix} = \underbrace{\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix} \qquad \text{for all } k = 1, \ldots, K,$$

or equivalently, in matrix form:

$$\underbrace{\begin{bmatrix} x'_1 & x'_2 & \cdots & x'_K \\ y'_1 & y'_2 & \cdots & y'_K \end{bmatrix}}_{\mathbf{L}'} = \underbrace{\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_K \\ y_1 & y_2 & \cdots & y_K \\ 1 & 1 & \cdots & 1 \end{bmatrix}}_{\mathbf{L}}$$

Here the matrices $\mathbf{L}$ and $\mathbf{L}'$ contain all the landmarks in images $\mathbf{X}$ and $\mathbf{X}'$.

In general, unless there exists a *perfect* linear alignment between $\mathbf{X}$ and $\mathbf{X}'$, there will exist no mapping $\mathbf{M}$ that satisfies this equality condition *for all* k. So instead we will look for mappings $\mathbf{M}$ that minimize the overall the distance between landmarks (x_k, y_k) and (x'_k, y'_k), i.e.,

$$\arg\min_{\mathbf{M}} \sum_{k=1}^{K} \left\| \begin{bmatrix} x'_k \\ y'_k \end{bmatrix} - \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix} \right\|_2^2,$$

or equivalently,

$$\arg\min_{\mathbf{M}} \quad \|\mathbf{L}' - \mathbf{M}\mathbf{L}\|_{\mathrm{F}}^2,$$

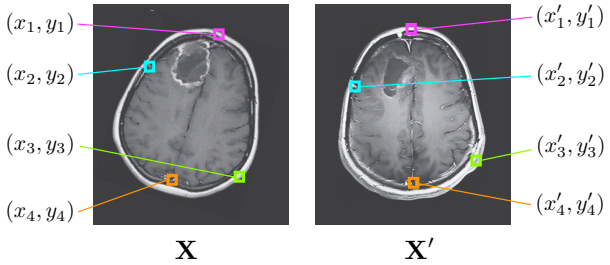


Figure 9.4: Landmark registration. Step 1: find *landmark* points (x_k, y_k) in $\mathbf{X}$ that correspond to points (x'_k, y'_k) in $\mathbf{X}'$.

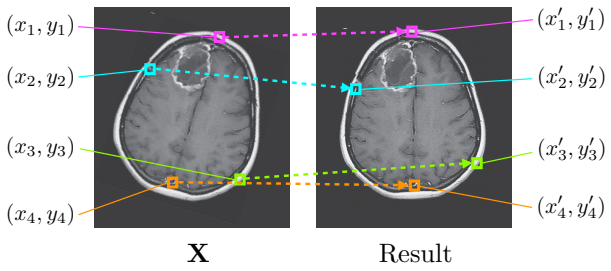


Figure 9.5: Landmark registration. Step 2: find mapping $\mathbf{M}$ that moves landmarks from positions (x_k, y_k) to positions (x'_k, y'_k), and apply such mapping to *all* pixels in $\mathbf{X}$.

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm, given by the square-root of the sum of squared entries in a matrix.

To find such $\mathbf{M}$, we follow the optimization 101 recipe: take derivative, set to zero, and solve for the variable of interest. To this end write:

$$\begin{aligned}
\arg\min_{\mathbf{M}} \ \|\mathbf{L}' - \mathbf{M}\mathbf{L}\|_{\mathrm{F}}^2 &= \arg\min_{\mathbf{M}} \ \mathsf{tr}[(\mathbf{L}' - \mathbf{M}\mathbf{L})^{\mathsf{T}}(\mathbf{L}' - \mathbf{M}\mathbf{L})] \\
&= \arg\min_{\mathbf{M}} \ \mathsf{tr}[\mathbf{L}'^{\mathsf{T}}\mathbf{L}' - 2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}' + \mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L}] \\
&= \arg\min_{\mathbf{M}} \ \mathsf{tr}[-2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}' + \mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L}],
\end{aligned}$$

where tr denotes the trace of a matrix (given by the sum of its diagonal entries); you can verify that the first equality is true by simply expanding the terms. The last equality follows because $\mathbf{L}'^{\mathsf{T}}\mathbf{L}'$ is a constant with respect to $\mathbf{M}$, so it can be ignored.

Next notice that $\mathbf{M}$ is a matrix, so taking its derivative is a bit tricky (not much, though). To learn more about how to take derivatives w.r.t. vectors and matrices, I recommend *Old and new matrix algebra useful for statistics* by Thomas P. Minka. Using the tricks therein, we have that the differential d is:

$$\begin{aligned}
\mathrm{d}\ \mathsf{tr}[-2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}' + \mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L}] &= \mathsf{tr}[-\mathrm{d}(2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}') + \mathrm{d}(\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L})] \\
&= \mathsf{tr}[-2\mathbf{L}^{\mathsf{T}}(\mathrm{d}\mathbf{M}^{\mathsf{T}})\mathbf{L}' + \mathbf{L}^{\mathsf{T}}\mathrm{d}(\mathbf{M}^{\mathsf{T}}\mathbf{M})\mathbf{L}] \\
&= \mathsf{tr}[-2\mathbf{L}^{\mathsf{T}}(\mathrm{d}\mathbf{M}^{\mathsf{T}})\mathbf{L}' + \mathbf{L}^{\mathsf{T}}(\mathrm{d}\mathbf{M}^{\mathsf{T}})\mathbf{M}\mathbf{L} + \mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}(\mathrm{d}\mathbf{M})\mathbf{L}] \\
&= \mathsf{tr}[-2(\mathrm{d}\mathbf{M}^{\mathsf{T}})\mathbf{L}'\mathbf{L}^{\mathsf{T}} + (\mathrm{d}\mathbf{M}^{\mathsf{T}})\mathbf{M}\mathbf{L}\mathbf{L}^{\mathsf{T}} + \mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}(\mathrm{d}\mathbf{M})] \\
&= \mathsf{tr}[-2\mathbf{L}\mathbf{L}'^{\mathsf{T}}(\mathrm{d}\mathbf{M}) + \mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}(\mathrm{d}\mathbf{M}) + \mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}(\mathrm{d}\mathbf{M})] \\
&= \mathsf{tr}[-2\mathbf{L}\mathbf{L}'^{\mathsf{T}}(\mathrm{d}\mathbf{M}) + 2\mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}(\mathrm{d}\mathbf{M})] \\
&= \mathsf{tr}[(-2\mathbf{L}\mathbf{L}'^{\mathsf{T}} + 2\mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}})(\mathrm{d}\mathbf{M})].
\end{aligned}$$

Hence we conclude that

$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{M}}\mathsf{tr}[-2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}' + \mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L}] = -2\mathbf{L}\mathbf{L}'^{\mathsf{T}} + 2\mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}.$$

Intuitively, we can think of $2\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{L}'$ as a constant ($\mathbf{L}\mathbf{L}'^{\mathsf{T}}$) times some scalar variable ($\mathbf{M}$), and we can similarly think of $\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{M}\mathbf{L}$ as a constant ($\mathbf{L}\mathbf{L}^{\mathsf{T}}$) times a squared scalar variable ($\mathbf{M}^{\mathsf{T}}\mathbf{M}$). Hence the derivative of the first term is simply the constant, and the derivative of the second term is 2 times the constant times the variable.

Finally, we set the derivative to zero and solve for the variable of interest, to obtain:

$$\begin{aligned} -2\mathbf{L}\mathbf{L}'^{\mathsf{T}} + 2\mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}} &= 0 \\ \mathbf{L}\mathbf{L}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}} &= \mathbf{L}\mathbf{L}'^{\mathsf{T}} \\ \mathbf{M}^{\mathsf{T}} &= (\mathbf{L}\mathbf{L}^{\mathsf{T}})^{-1}\mathbf{L}\mathbf{L}'^{\mathsf{T}} \\ \mathbf{M} &= \mathbf{L}'\mathbf{L}^{\mathsf{T}}(\mathbf{L}\mathbf{L}^{\mathsf{T}})^{-1}. \end{aligned}$$

Notice that $\mathbf{L}\mathbf{L}^{\mathsf{T}}$ must be invertible for this to work, which will be generally the case as long as there are at least $K = 3$ linearly independent landmarks in $\mathbf{L}$.

9.4.1 Algorithm

Hence, landmark image registration can be summarized as follows:

- Arrange all landmark points in $\mathbf{X}$ and $\mathbf{X}'$ into matrices $\mathbf{L}$ and $\mathbf{L}'$:

$$\mathbf{L} = \begin{bmatrix} x_1 & x_2 & \cdots & x_K \\ y_1 & y_2 & \cdots & y_K \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \qquad \mathbf{L}' = \begin{bmatrix} x'_1 & x'_2 & \cdots & x'_K \\ y'_1 & y'_2 & \cdots & y'_K \end{bmatrix}.$$

- Compute $\mathbf{M} = \mathbf{L}'\mathbf{L}^{\mathsf{T}}(\mathbf{L}\mathbf{L}^{\mathsf{T}})^{-1}$.
- Relocate *all* pixels (x, y) in $\mathbf{X}$ into new positions (x', y') given by:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$