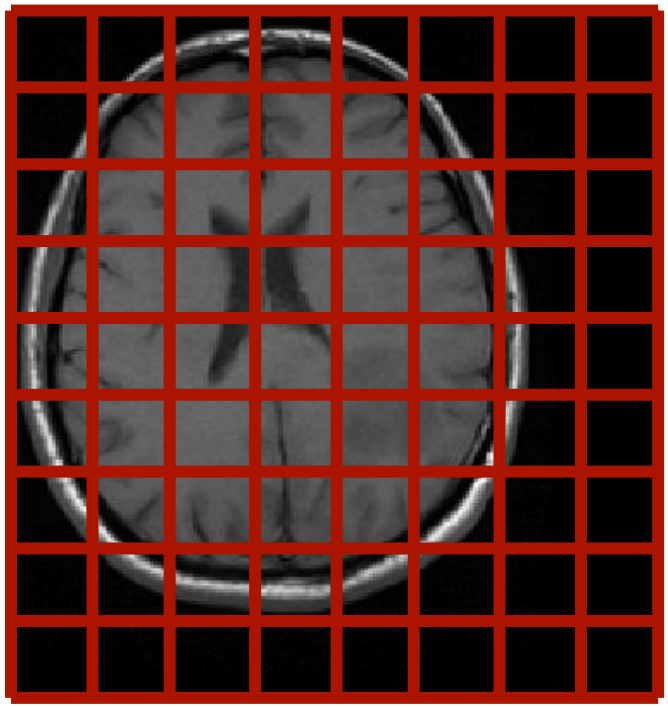
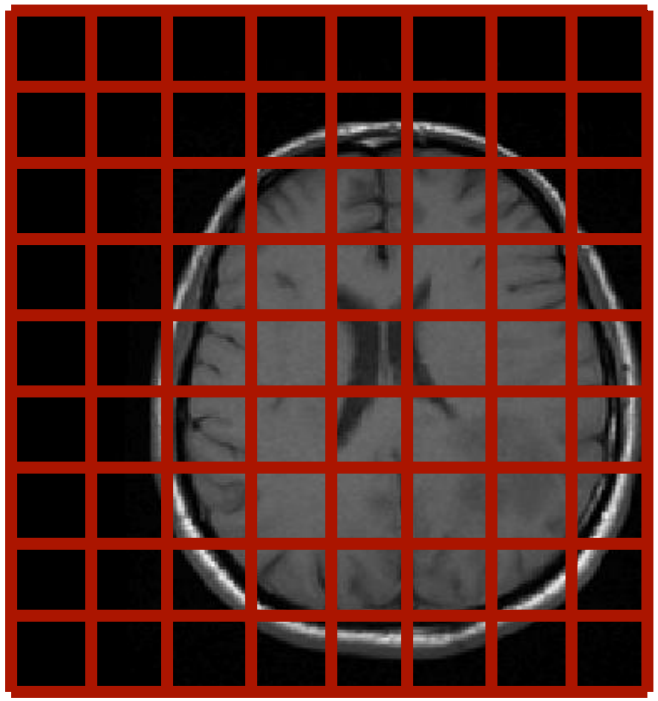
Image registration

rigid and affine





Literature: L. Romero, “A tutorial on parametric image registration”, 2007

Overview

Block 2: Image processing

Registration	Lecture 1:	Basics of parametric image registration
	Lecture 2:	Multi-modality registration using Mutual Information
	Lecture 3+4:	Deformable registration (Greg Sharp)

Image registration - Motivation

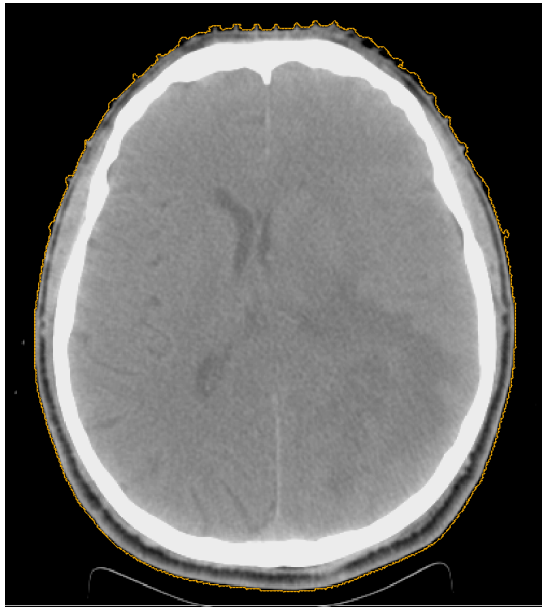
1. **Multiple imaging modalities**

   different images (CT, MR, PET, …) show different features

Motivation

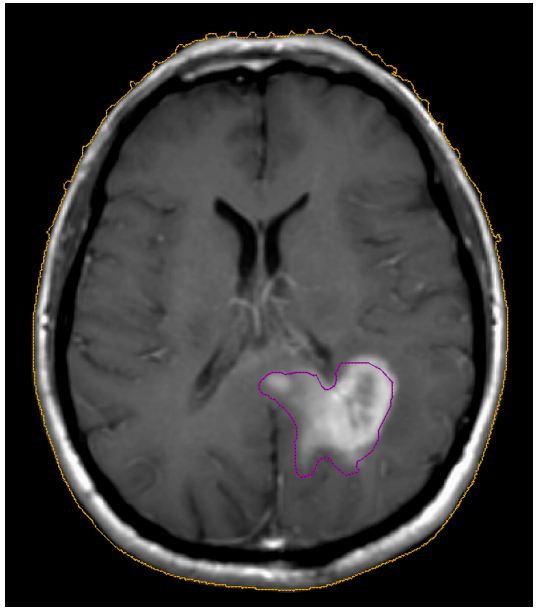
1. Multiple imaging modalities

different images (CT, MR, PET, …) show different features



CT: bone, soft tissue, air

absorption coefficients for radiotherapy dose calculation



T1-MRI: tumor, ventricles, white matter

Motivation

1. **Multiple imaging modalities**

   different images (CT, MR, PET, …) show different features

2. **Images from different time points**

   examples: planning CT + cone beam CT in treatment position

   diagnostic image + follow-up images

Motivation

1. **Multiple imaging modalities**

   different images (CT, MR, PET, …) show different features

2. **Images from different time points**

   examples: planning CT + cone beam CT in treatment position

   diagnostic image + follow-up images

3. **Atlas registration**

   for atlas based segmentation

Overview

Today: Intra-modality registration

e.g. CT to CT (Hounsfield number is quantitative)

Formulate as optimization problem

Problem formulation

Image transformation

Gradient calculation

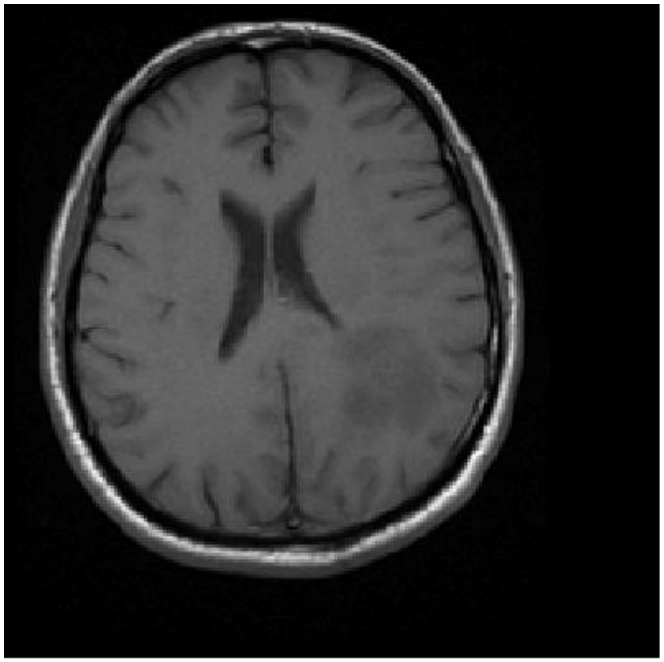
Next lecture: Multi-modality registration

Mutual information

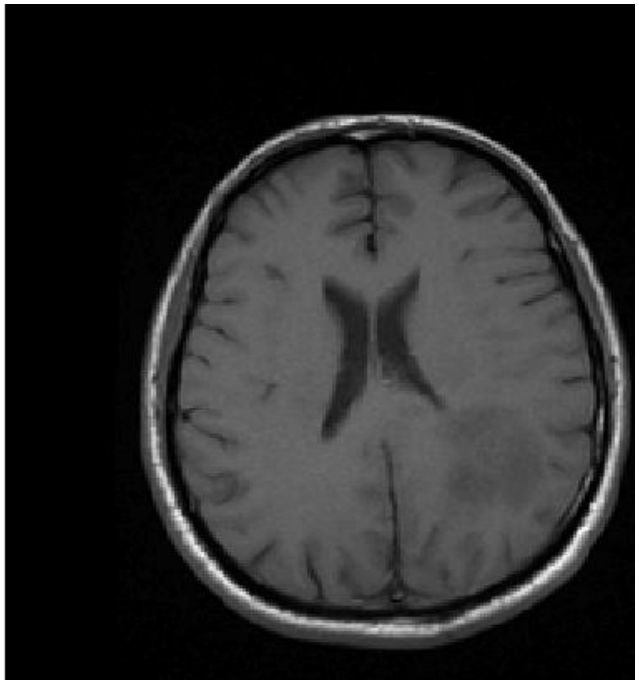
Problem formulation

Goal: find a transformation that maps every voxel in one image to the corresponding voxel in the reference image

Image

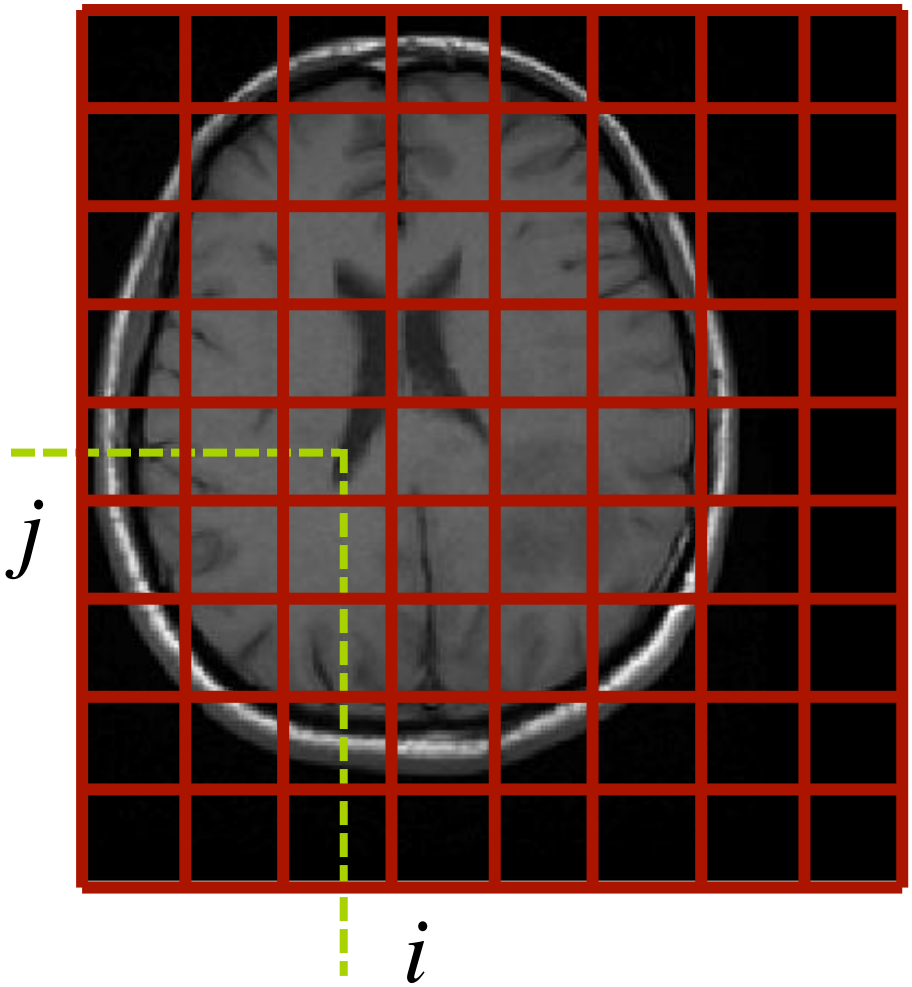


Reference Image

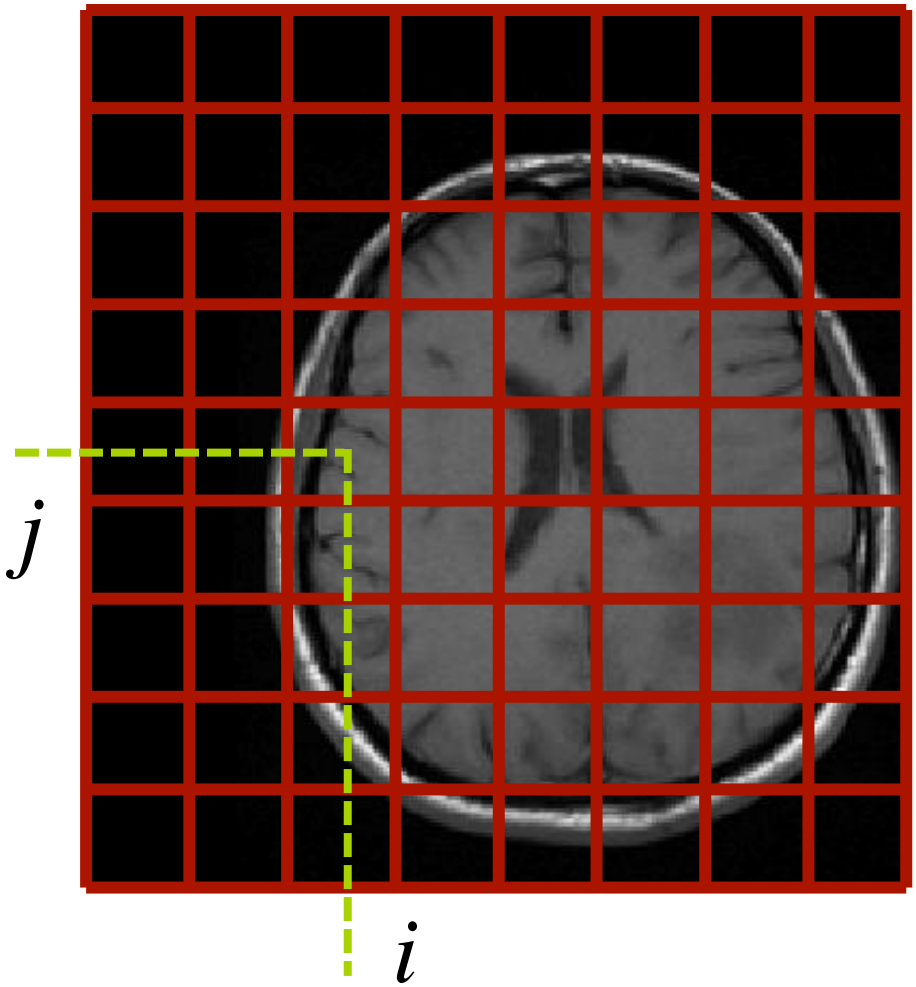


Problem formulation

Image $I(i,j)$

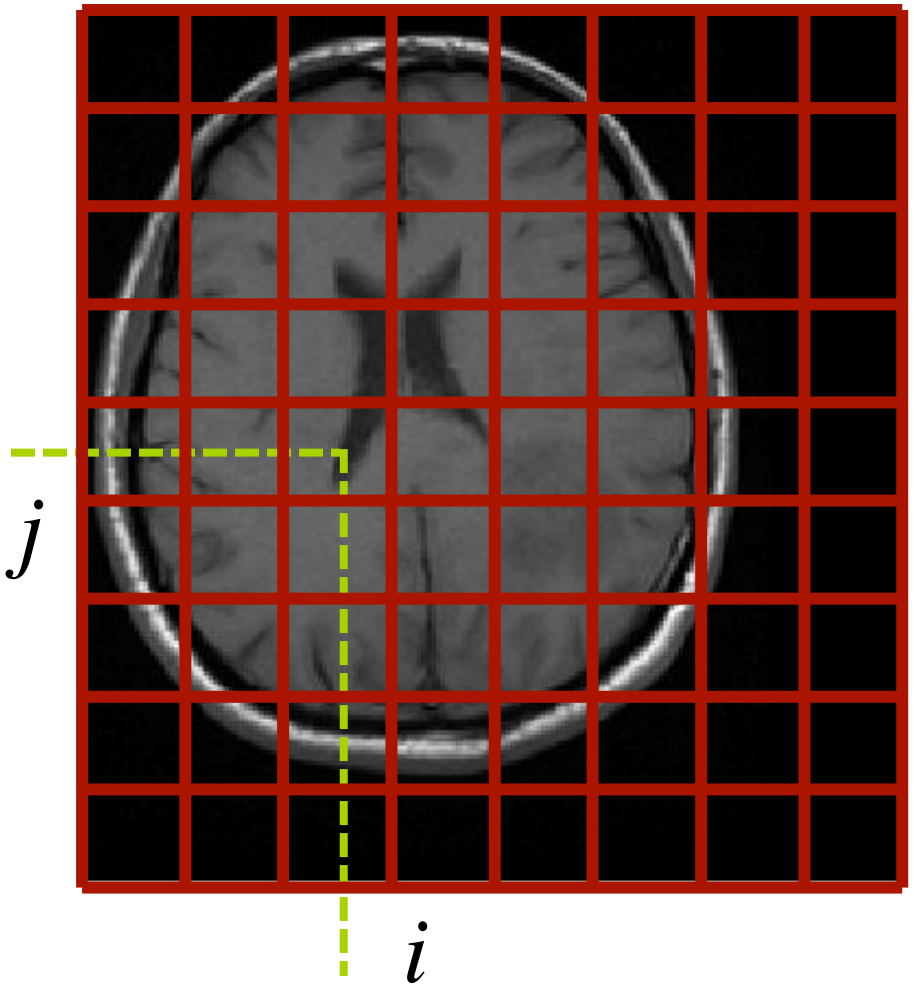


Reference Image $I_R(i,j)$

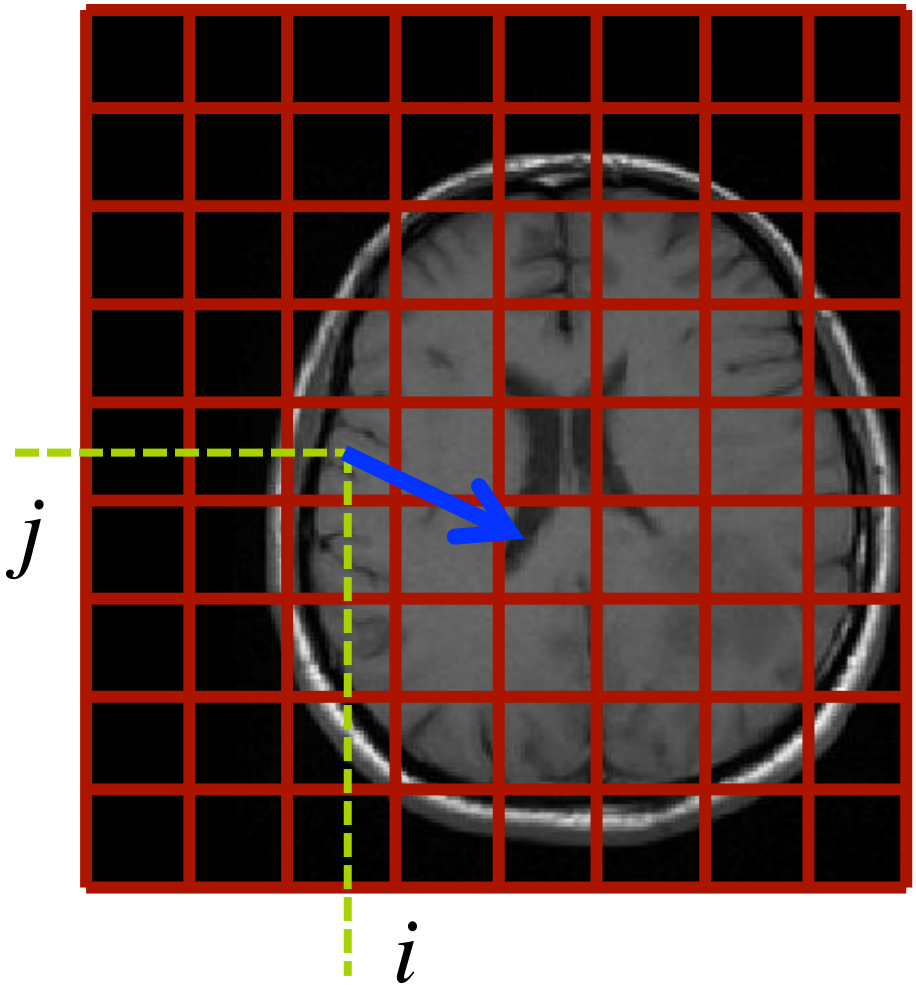


Problem formulation

Image



Reference Image



Problem formulation

Consider the image as a function in 3D (here 2D)

$$I(x, y)$$ (scalar, real valued)

We have a discrete representation of the function

$$I(i, j)$$ (evaluated at voxel (i,j))

Image transformation means

- coordinate transformation

$$I_R(i, j) \sim I\left(x = f_x(i, j), y = f_y(i, j)\right)$$

- not changing the “image function” itself

Image transformation

$$\begin{pmatrix} x \\ y \end{pmatrix} = f(i, j; \Theta) = \begin{pmatrix} f_x(i, j; \Theta) \\ f_y(i, j; \Theta) \end{pmatrix}$$

Image transformation function evaluated at (i, j)

Point in transformed image

Image transformation

$$\begin{pmatrix} x \\ y \end{pmatrix} = f\left(i, j; \Theta\right) = \begin{pmatrix} x(i, j; \Theta) \\ y(i, j; \Theta) \end{pmatrix}$$

Image transformation function evaluated at (i, j)

Point in the image

Transformation function

Specification of the transformation function f

Rigid / Affine registration

- few parameters
- global representation of the transformation

Deformable registration

- many parameters
- local representation of the transformation

Rigid registration

Angles and length of line segments are preserved

- translation
- rotation

In 2D:
- 2 parameters for translation
- 1 angle for rotation

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

In 3D:
- 3 parameters for translation
- 3 angles for rotation

Rigid registration

Angles and lengths of line segments are preserved

- translation
- rotation

In 2D:
- 2 parameters for translation
- 1 angle for rotation

Homogeneous coordinates: (single matrix multiplication)

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi & t_x \\ \sin\varphi & \cos\varphi & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}$$

Affine registration

Most general transformation that preserves parallelism between lines

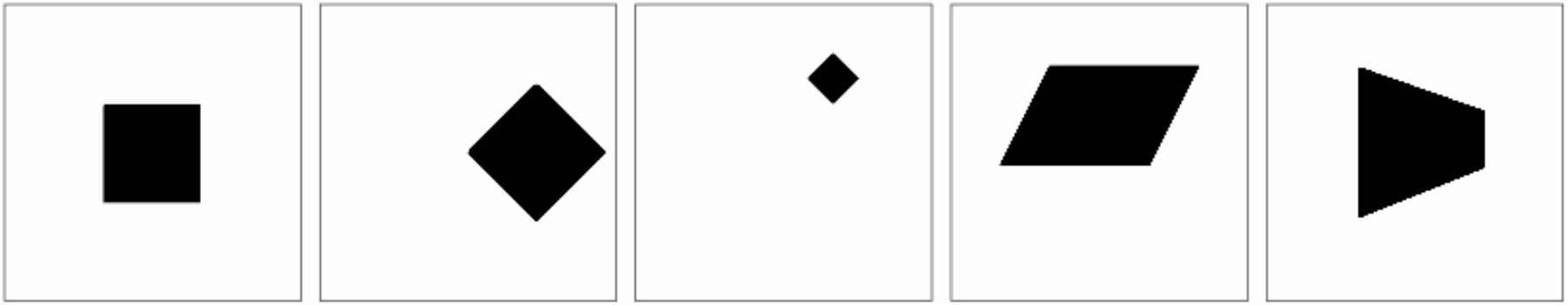
- translation 2
- rotation 1
- scaling 2
- shearing 1

2D: 6 parameters

3D: 12 parameters

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}$$

Basic transformations



translation rotation similarity affine projective

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ e & f & 1 \end{pmatrix} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}$$

Optimization problem

Determining the transformation parameters …

"make transformed image similar to the reference image"

Formulate as optimization problem

$$\underset{\Theta}{\text{minimize}} \quad C = \sum_{i,j} \left[I\left(x(i,j;\Theta), y(i,j;\Theta)\right) - I_R\left(i,j\right) \right]^2$$

$$C = \sum_{i,j} g\left(e_{ij}\right) \qquad g\left(e_{ij}\right) = e_{ij}^2$$

Optimization algorithm

Most generic algorithm: gradient descent

$$\Theta^{t+1} \leftarrow \Theta^{t} - \alpha \nabla_{\Theta} C$$

$$\nabla_{\Theta} C = \begin{pmatrix} \frac{\partial C}{\partial \Theta_1} \\ \vdots \\ \frac{\partial C}{\partial \Theta_n} \end{pmatrix}$$

Note: calculating the gradient is also the basis for more advanced algorithms

e.g. Quasi-Newton methods

Calculating derivatives

Calculating derivatives (chain rule)

$$\frac{\partial C}{\partial \Theta_k} = \sum_{i,j} \frac{\partial C}{\partial e_{ij}} \frac{\partial e_{ij}}{\partial \Theta_k}$$

"how much does the error in voxel (i,j) change if the transformation parameters change"

influence function

"how much does the objective function change if the error in voxel (i,j) changes"

$$= 2e_{ij} = 2\left(I\left(x(i,j;\Theta), y(i,j;\Theta)\right) - I_R\left(i,j\right)\right)$$

Calculating derivatives

Calculating derivatives

$$\frac{\partial e_{ij}}{\partial \Theta_k} = \frac{\partial}{\partial \Theta_k}\Big(I\big(x(i,j;\Theta), y(i,j;\Theta)\big) - I_R(i,j)\Big)$$

independent of Θ

$$\begin{pmatrix} \dfrac{\partial x(i,j)}{\partial \Theta_k} & \dfrac{\partial y(i,j)}{\partial \Theta_k} \end{pmatrix} \begin{pmatrix} \dfrac{\partial I\big(x(i,j;\Theta), y(i,j;\Theta)\big)}{\partial x} \\ \dfrac{\partial I\big(x(i,j;\Theta), y(i,j;\Theta)\big)}{\partial y} \end{pmatrix}$$

(chain rule in higher dimensions)

Calculating derivatives

$$\frac{\partial e_{ij}}{\partial \Theta_k} = \begin{pmatrix} \frac{\partial x(i,j)}{\partial \Theta_k} & \frac{\partial y(i,j)}{\partial \Theta_k} \end{pmatrix} \begin{pmatrix} \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial x} \\ \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial y} \end{pmatrix}$$

Coordinate matrix model

"how much does the point (x,y) change when changing the transformation parameters"

Image gradient

"how much does the image change at point (x,y)"

Calculating derivatives

$$\nabla_{\Theta} C = \sum_{i,j} \frac{\partial C}{\partial e_{ij}} \nabla_{\Theta} e_{ij}$$

$$\nabla_{\Theta} e_{ij} = \begin{pmatrix} \frac{\partial x(i,j)}{\partial \Theta_1} & \frac{\partial y(i,j)}{\partial \Theta_1} \\ \vdots & \vdots \\ \frac{\partial x(i,j)}{\partial \Theta_K} & \frac{\partial y(i,j)}{\partial \Theta_K} \end{pmatrix} \begin{pmatrix} \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial x} \\ \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial y} \end{pmatrix}$$

$K \times 2$ Matrix (K=#parameters)

Coordinate matrix model

Example: affine transformation

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \begin{pmatrix} \Theta_1 & \Theta_2 & \Theta_3 \\ \Theta_4 & \Theta_5 & \Theta_6 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \frac{\partial x(i,j)}{\partial \Theta_1} & \frac{\partial y(i,j)}{\partial \Theta_1} \\ \vdots & \vdots \\ \frac{\partial x(i,j)}{\partial \Theta_6} & \frac{\partial y(i,j)}{\partial \Theta_6} \end{pmatrix} = \begin{pmatrix} i & 0 \\ j & 0 \\ 1 & 0 \\ 0 & i \\ 0 & j \\ 0 & 1 \end{pmatrix}$$

Calculating image gradients

Image is given at discrete positions (i,j)

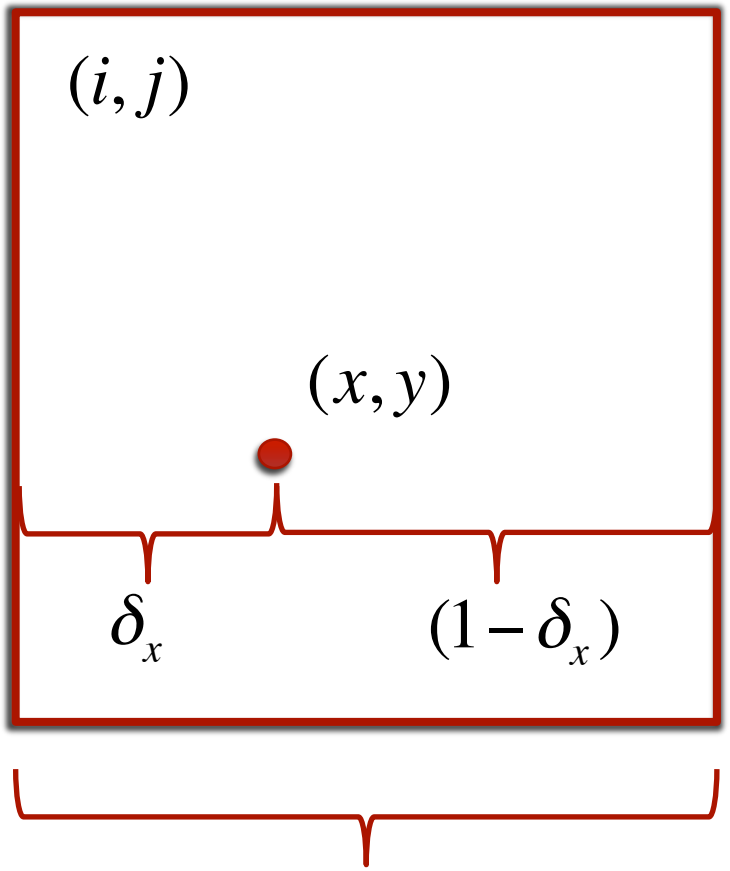
But, registration requires evaluation at (x,y)

$$I\left(x(i,j;\Theta), y(i,j;\Theta)\right)$$

Gradient at (x,y)

$$\begin{pmatrix} \dfrac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial x} \\ \dfrac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial y} \end{pmatrix}$$

Bilinear interpolation



$$I(x,y) = I(i+\delta_x\ ,\ j+\delta_y)$$

$$= \delta_x I(i+1, j+\delta_y) + (1-\delta_x) I(i, j+\delta_y)$$

$$\delta_y I(i+1, j+1) + (1-\delta_y) I(i+1, j)$$

Calculating image derivatives

Simplest method: finite differences

$$\frac{\partial I(i,j)}{\partial i} \approx \frac{I(i+1,j) - I(i-1,j)}{2}$$

Main problem:

- Sensitive to noise

potentially better: Derivatives of Gaussian filters

Calculating image derivatives

Consider smoothed image:

$$I_S(x,y) = \iint_{\Re} I(x',y')g(x-x')g(y-y')\,dx'\,dy'$$

Gaussian

Derivative:

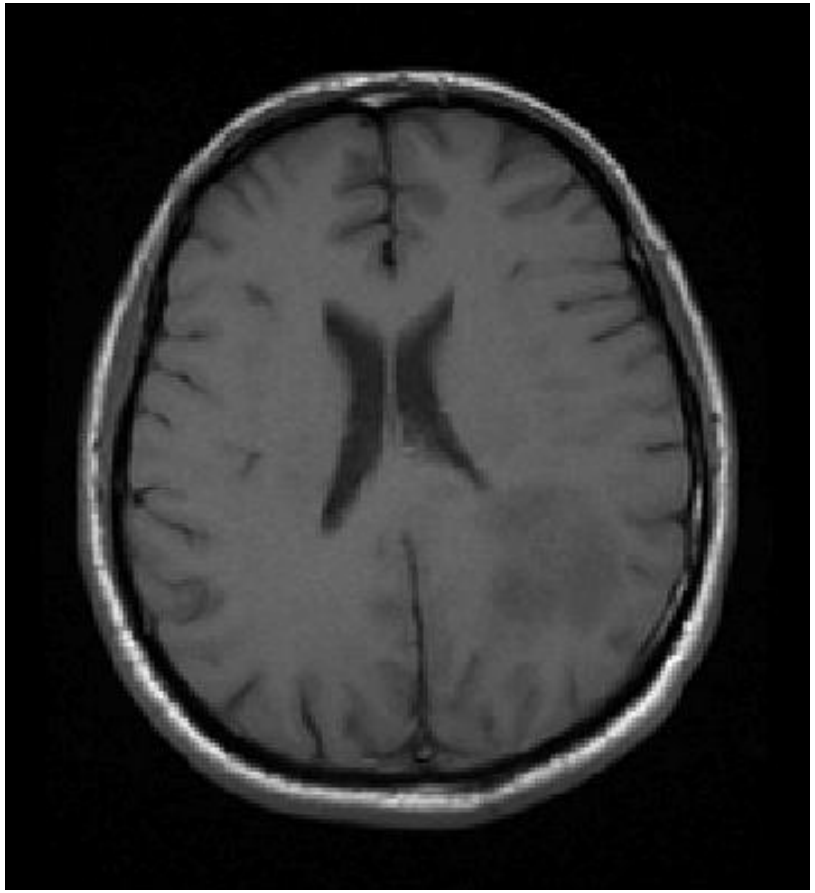
$$\frac{\partial I_S(x,y)}{\partial x} = \frac{\partial}{\partial x}\iint_{\Re} I(x',y')g(x-x')g(y-y')\,dx'\,dy'$$

$$= \iint_{\Re} I(x',y')g'(x-x')g(y-y')\,dx'\,dy'$$

Derivative of the Gaussian

$$\frac{\partial I(i,j)}{\partial i} \approx \sum_{i'}\sum_{j'} I(i',j')g'(i-i')g(j-j') \quad \text{(discrete version)}$$

Image gradient

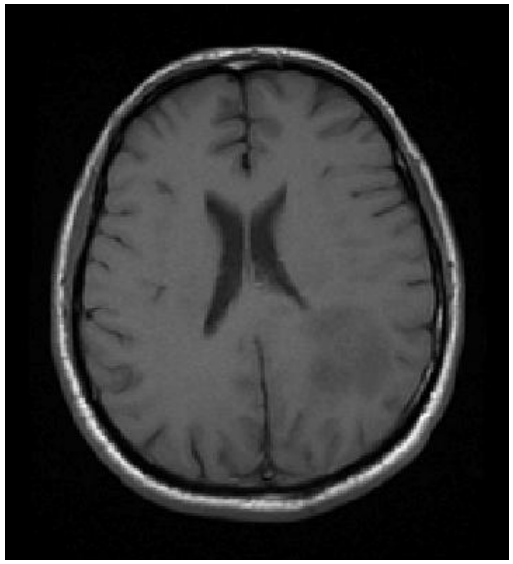


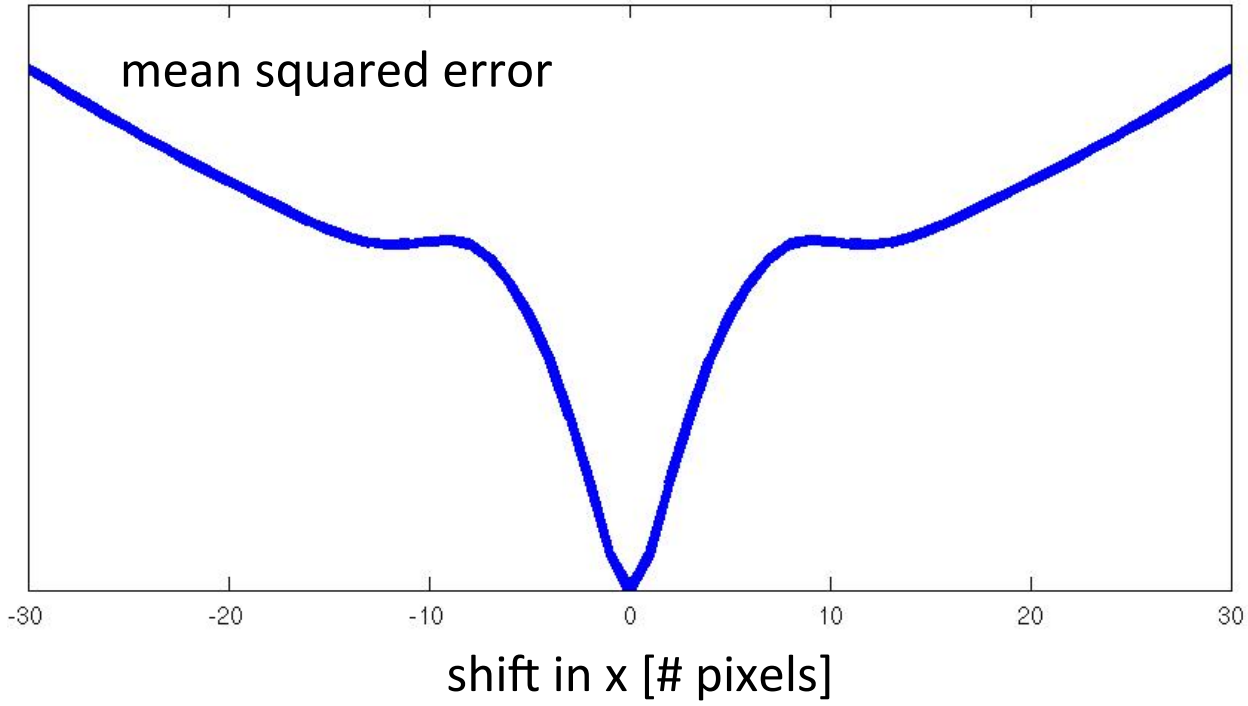
Image

y-component of gradient

Local minima

Mean Squared Difference for shifted image

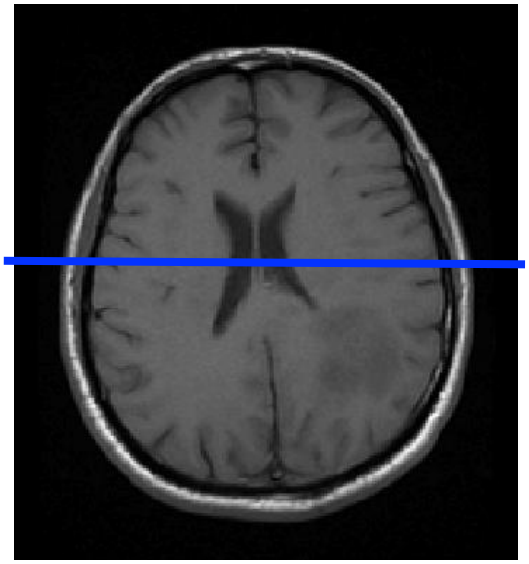


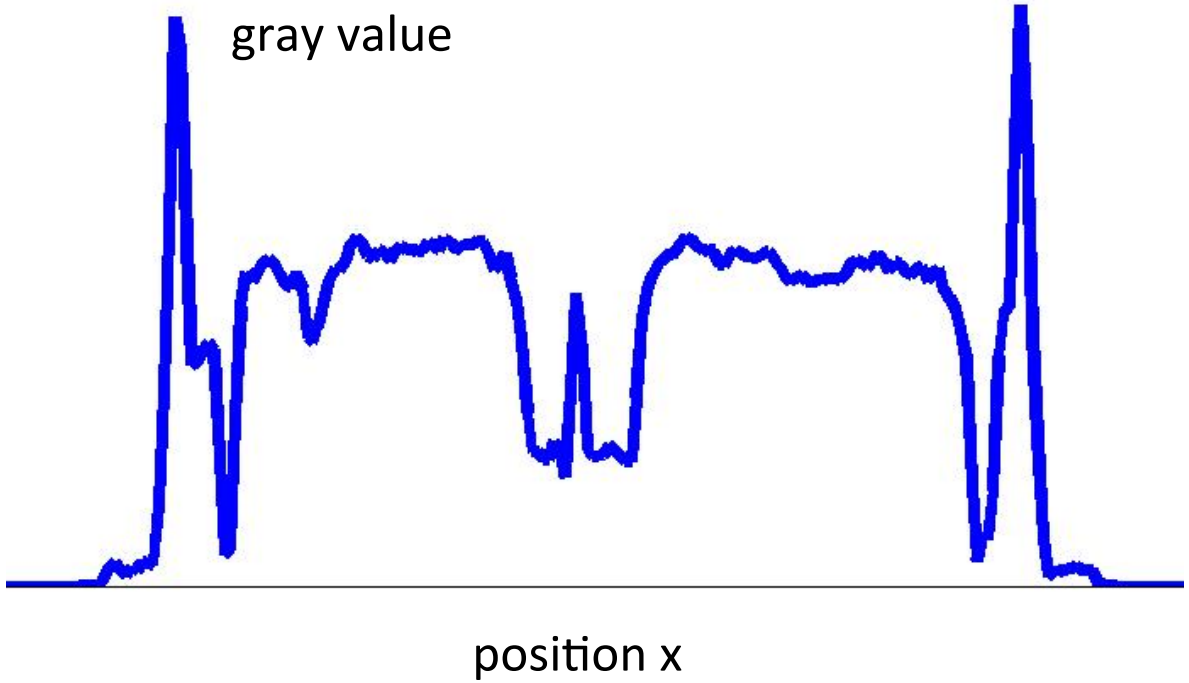


➔ Local minima occur

Local minima

Mean Squared Difference for shifted image





The image is a non-convex function

Summary

Topic: Parametric image registration

explicit representation of coordinate transformation

(rigid, affine)

So far: Intra-modality registration

Minimize difference of reference and transformed image

Use chain rule to calculate gradients

- influence function
- image gradient
- coordinate matrix model

Summary

Quadratic objective: $\underset{\Theta}{\text{minimize}} \sum_{i,j} \left[I\left(x(i,j;\Theta), y(i,j;\Theta)\right) - I_R(i,j) \right]^2$

Gradient evaluation: $\frac{\partial C}{\partial \Theta_k} = \sum_{i,j} \frac{\partial C}{\partial e_{ij}} \frac{\partial e_{ij}}{\partial \Theta_k}$

Influence function

$$\frac{\partial e_{ij}}{\partial \Theta_k} = \begin{pmatrix} \frac{\partial x(i,j)}{\partial \Theta_k} & \frac{\partial y(i,j)}{\partial \Theta_k} \end{pmatrix} \begin{pmatrix} \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial x} \\ \frac{\partial I\left(x(i,j;\Theta), y(i,j;\Theta)\right)}{\partial y} \end{pmatrix}$$

Coordinate matrix model

Image gradient