



# Point-Based Registration and Optimization

Dr. Martin Groher

Computer Aided Medical Procedures (CAMP),  
Technische Universität München, Germany

---

## Organizational Issues

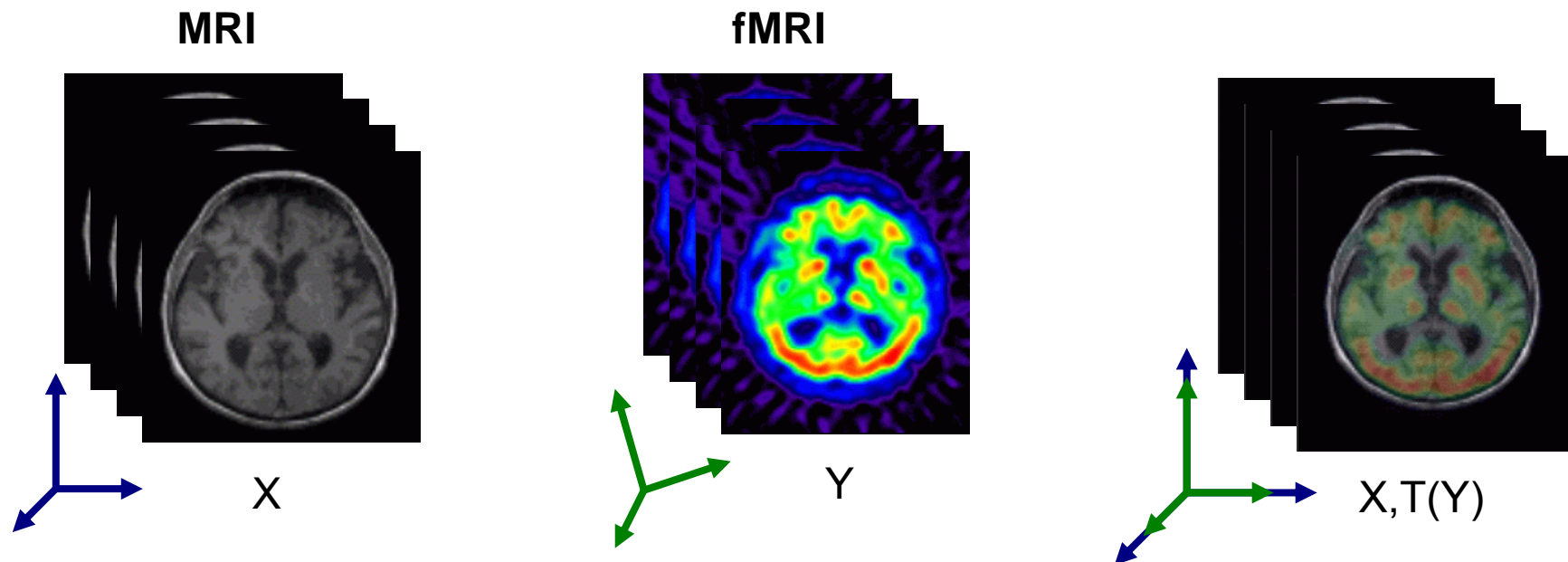
- List of participants:  
**please fill out all missing information/register if your name does not appear on the list**
  
- Attestation:  
**Shift from Tuesday, June 9<sup>th</sup>, 2.30pm to Monday, June 8<sup>th</sup>, 5.00pm**



## Outline, Point-Based Registration

- Point-Based Registration
  - Point-Based Registration without correspondences: Principal Components Analysis (PCA)
  - Point-Based Registration with initial estimate: Iterative Closest Point (ICP)
-

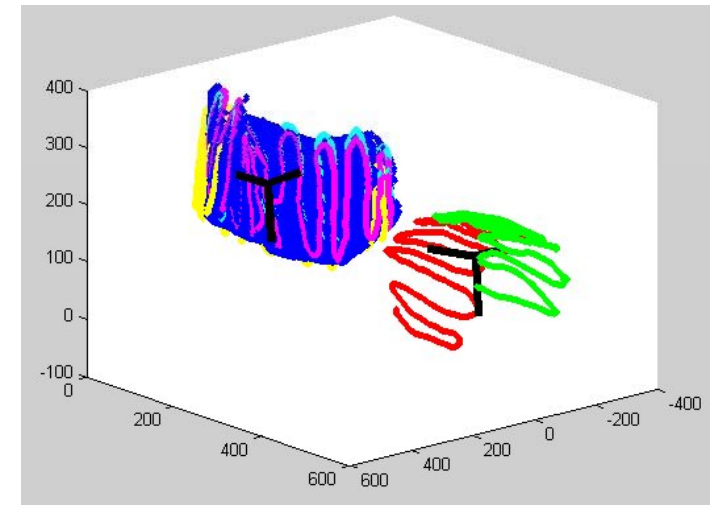
# Registration



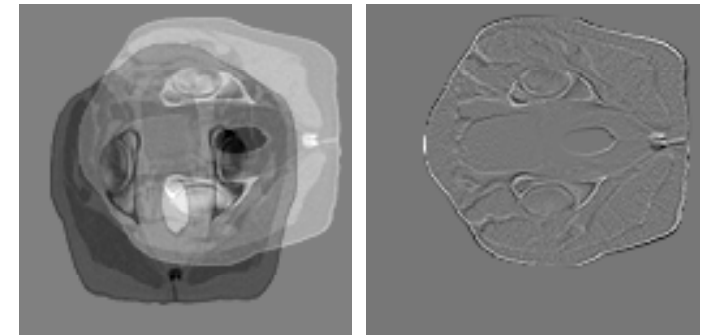
- Define Transformation  $T$  on one of the images
- Compare  $X$ ,  $T(Y)$  using either features or the full image content
- Refine  $T$  and repeat, until convergence criteria reached.

## Registration Overview

- Feature-based methods
  - Point/line/curve/surface extraction
  - Registration of extracted entities only
  - Problem becomes one of **correspondences** and **distance**:
    - Corresponding Data available: Umeyama
    - No corresponding data available: PCA and ICP
- Intensity-based methods
  - Registration on pixel intensities only
  - Problem becomes one of **interpolation** and **similarity**:
    - How to get the right values to compare: Image Warping
    - How to compare the values in a meaningful way: Similarity Measures



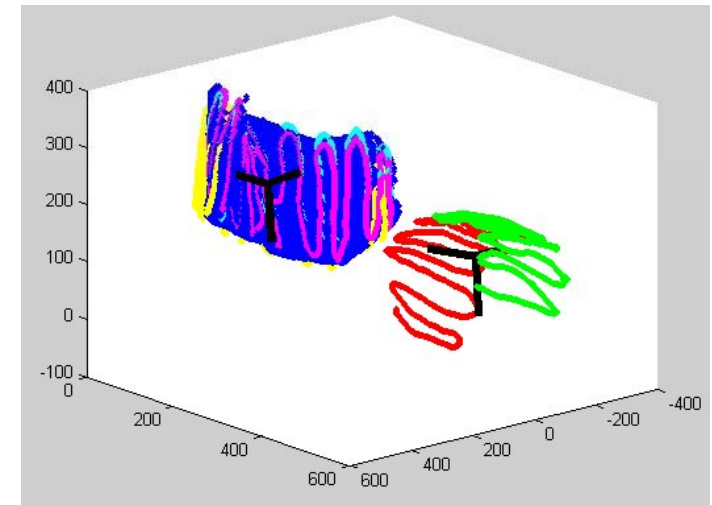
Registration of point sets, see exercise



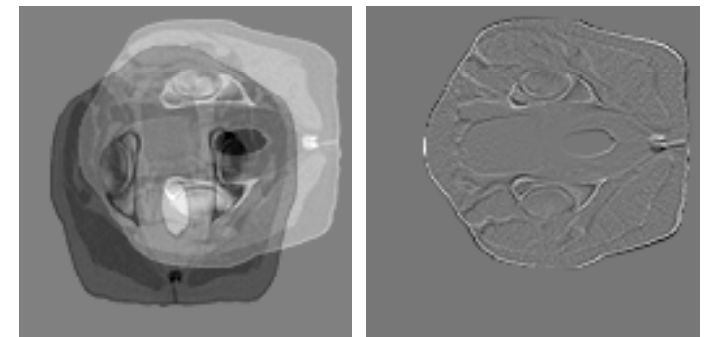
Registration of 2 CT slices, difference images

## Registration Overview

- Feature-based methods
  - Point/line/curve/surface extraction
  - Registration of extracted entities only
  - Problem becomes one of **correspondences** and **distance**:
    - Corresponding Data available: Umeyama
    - No corresponding data available: PCA and ICP
- Intensity-based methods
  - Registration on pixel intensities only
  - Problem becomes one of **interpolation** and **similarity**:
    - How to get the right values to compare: Image Warping
    - How to compare the values in a meaningful way: Similarity Measures



Registration of point sets, see exercise



Registration of 2 CT slices, difference images

## Corresponding 3D Point Sets – Problem

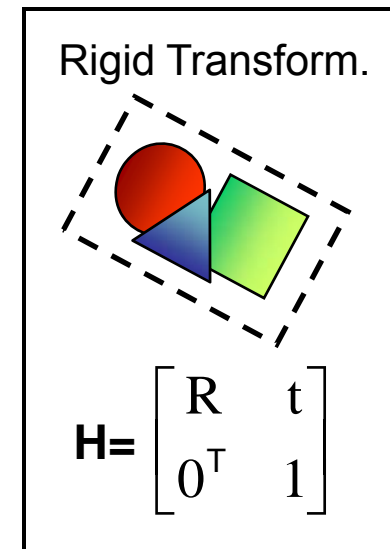
- Given: Two corresponding point sets  $\{x_i\}, \{y_i\}$  in 3D
- Task: Find the rigid motion  $H$  between the point sets:

$$\forall_i : y_i = Hx_i$$
$$Y_{4 \times N} = H_{4 \times 4} X_{4 \times N}$$

where  $H$  is a rigid 6-DOF transformation  
→ known as *Procrustes Problem*

- In inhomogeneous coordinates:

$$\forall_i : y_i = Rx_i + t$$



## Corresponding 3D Point Sets – Umeyama Method

- Demean the point sets:

$$\begin{aligned}x'_i &= x_i - \bar{x} & y'_i &= y_i - \bar{y} \\ \forall_i : y'_i &= Rx'_i & Y'_{3 \times N} &= RX'_{3 \times N}\end{aligned}$$

- Compute rotation with SVD of correlation matrix:

$$\begin{aligned}svd(X'Y'^T) &= UDV^T \\ R &= VU^T\end{aligned}$$

- Compute translation:

$$t = \bar{y} - R\bar{x}$$

## Corresponding 3D Point Sets (2)

- Singular Value Decomposition:

$$A_{m \times n} = U_{m \times m} D_{m \times n} V_{n \times n}^T$$

$$V^T V = I$$

$$U^T U = I$$

- Special case for symmetric matrix:

$$X X^T = U D U^T$$

- Applied to our rotation problem:

$$Y = R X \rightarrow Y^T = X^T R^T \rightarrow X Y^T = X X^T R^T = U D \overbrace{U^T}^{V^T} R^T$$

$$\rightarrow R = V U^T$$

## Corresponding 3D Point Sets - Summary

- Good method if correspondences are known
  - If no corresponding points are available (only two point clouds) method fails
  - combination PCA-ICP
  - PCA: use center of masses and principal components of point clouds for initial estimate of  $R, t$
  - ICP: iterative solution to registration problem from initial estimate
-

## 3D Point Sets w/o Correspondences - PCA

- Demean the point sets:

$$x'_i = x_i - \bar{x} \quad y'_i = y_i - \bar{y}$$

$$\forall_i : y'_i = R x'_i \quad Y'_{3 \times N} = R X'_{3 \times N}$$

- Compute principal components of both point sets (SVD of covariance matrices):

$$X'X'^T = \overbrace{U_1}^{R_X} D_1 U_1^T \quad Y'Y'^T = \overbrace{U_2}^{R_Y} D_2 U_2^T$$

$$\rightarrow R = R_Y R_X^T$$

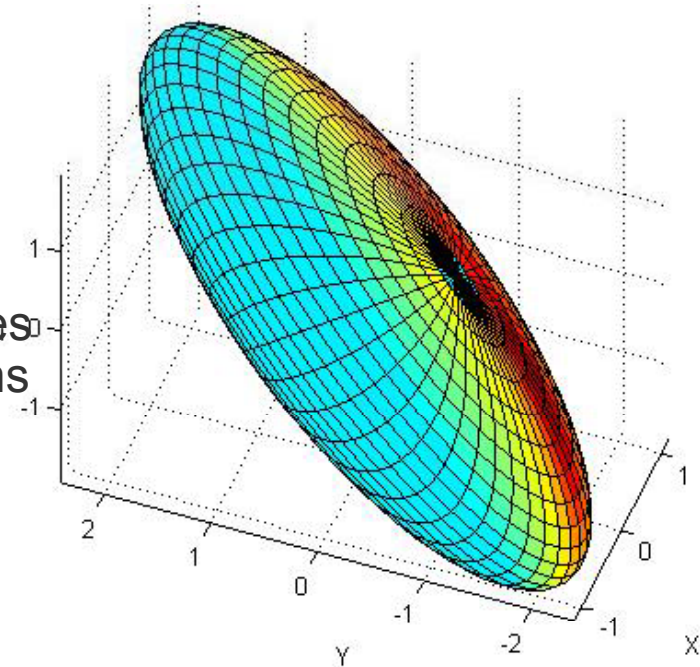
Different from  
Umeyama method

- Compute translation:

$$t = \bar{y} - R\bar{x}$$

## Registration based on PCA - Summary

- Pro: Analytical solution: Fast
- Pro: No correspondences needed
- Con: Registration not unique!  
Principal components with their magnitudes form ellipsoid. An interpretation as rotations leaves 4 directions of the axes without altering the shape of the ellipsoid
- Con: Registration not very accurate
  - $\rightarrow$  can be used as initialization for another registration algorithm, e.g. Iterative Closest Point (ICP)



## Point Based Registration - Summary

- Accurate if corresponding information available
- If no correspondences are given only rough initial estimate via PCA
- How to improve the estimate?

- → iterative methods
- → robust methods
- → using of initial estimate



**Iterative Closest Point**

[Besl, Mccay '92], [Zhang '92]

## Iterative Closest Point

Algorithm Outline:

INPUT: Two spatial entities (point sets, surfaces, curves)  $E_1, E_2$

OUTPUT: Transformation  $R, t$ , such that  $E_2 = RE_1 + t$

INIT:  $\hat{E}_1 := E_1$

1.  $\forall p \in E_2$  : Determine the closest points in  $\hat{E}_1 \rightarrow$  Match  $M_{E_1 E_2}$
2. Update the Matching  $M_{E_1 E_2}$  using a statistical analysis to get rid of outliers  $\rightarrow$  Match  $\hat{M}_{E_1 E_2}$
3. Assume the points in  $\hat{M}_{E_1 E_2}$  correspond and estimate the motion  $R, t$  using some linear analytical method, e.g. the Umeyama method
4. Apply the motion  $R, t$  to all points in  $E_1$  and repeat until convergence

1.  $\forall p \in E_2$  : Determine the closest points in  $\hat{E}_1 \rightarrow$  Match  $M_{E_1 E_2}$

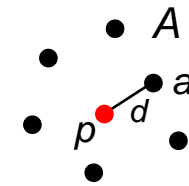
- Euclidian distance between point  $p_1=(x_1, y_1, z_1)$  and  $p_2=(x_2, y_2, z_2)$

$$d(p_1, p_2) = \|p_1 - p_2\| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

- Point to basic geometric entity distance

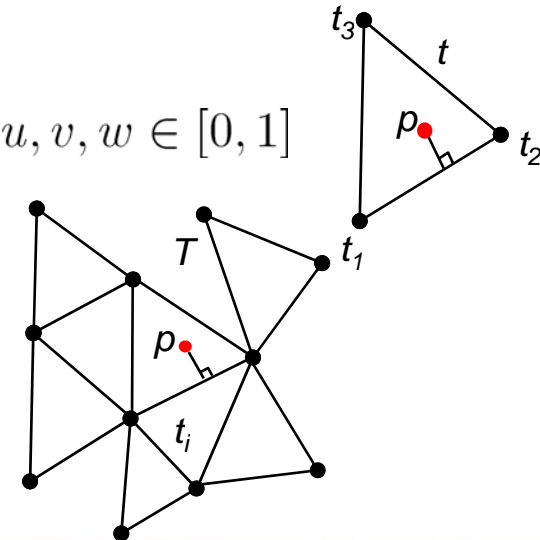
- point set  $A=\{a_i\}, i=1, \dots, N_A$

$$d(p, A) = \min_{I \in \{1, \dots, N_A\}} d(p, a_i)$$



- triangle  $t$  defined by  $t_1, t_2,$  and  $t_3$

$$d(p, t) = \min_{u+v+w=1} \|ut_1 + vt_2 + wt_3 - p\|, u, v, w \in [0, 1]$$



- triangle set  $T=\{t_i\}, i=1, \dots, N_T$

$$d(p, T) = \min_{I \in \{1, \dots, N_T\}} d(p, t_i)$$

## 2. Update the Matching $M_{E_1 E_2}$ using a statistical analysis to get rid of outliers $\rightarrow$ Match $\hat{M}_{E_1 E_2}$

- Determine the mean and sample deviation of all distances:

$$\mu = \frac{1}{N} \sum_{i=1}^N d_i \quad \sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (d_i - \mu)^2}$$

- Depending on these values, reset the distance threshold for outlier detection

if  $\mu < D$

Reg good

$$D_{max}^I = \mu + 3\sigma$$

else if  $\mu < 3D$

Reg still good

$$D_{max}^I = \mu + 2\sigma$$

else if  $\mu < 6D$

Reg not too bad

$$D_{max}^I = \mu + \text{sigma}$$

else

Reg really bad

$$D_{max}^I = \text{median of all distances}$$

endif

- Reject all outliers from the previously acquired set  $M_{E_1 E_2}$

**2. Update the Matching  $M_{E_1 E_2}$  using a statistical analysis to get rid of outliers  $\rightarrow$  Match  $\hat{M}_{E_1 E_2}$**

- How to choose the distance threshold  $D$ ?  
 $\rightarrow$  heuristics, i.e. play around 😊
- The value of  $D$  is crucial:
  - $D$  very small: maybe no initial transformation estimate (all point correspondences are discarded as outliers)
  - $D$  very big: maybe convergence against a wrong transformation (too many outliers considered as good matches)
  - Find better solution: Depends on the shape of the input data (curve, surface, point cloud, etc.)

**3. Assume the points in  $\hat{M}_{E_1 E_2}$  correspond and estimate the motion  $R, t$  using some linear analytical method, e.g. the Umeyama method**

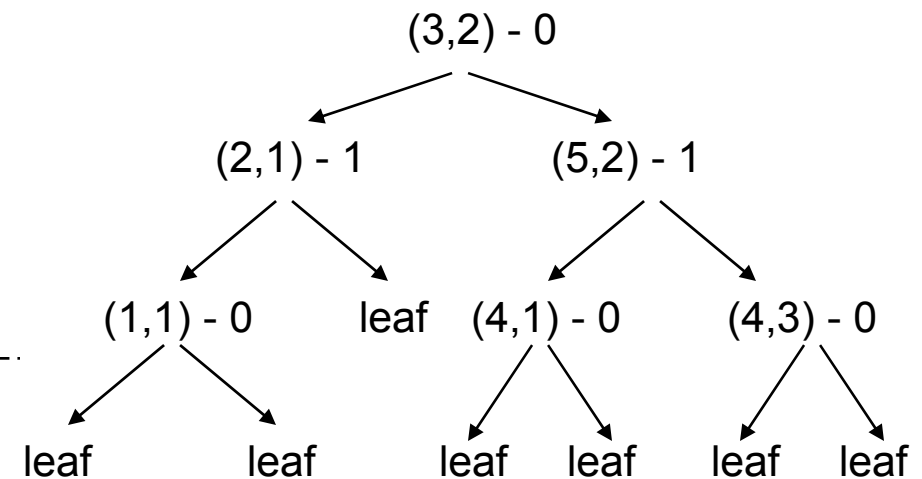
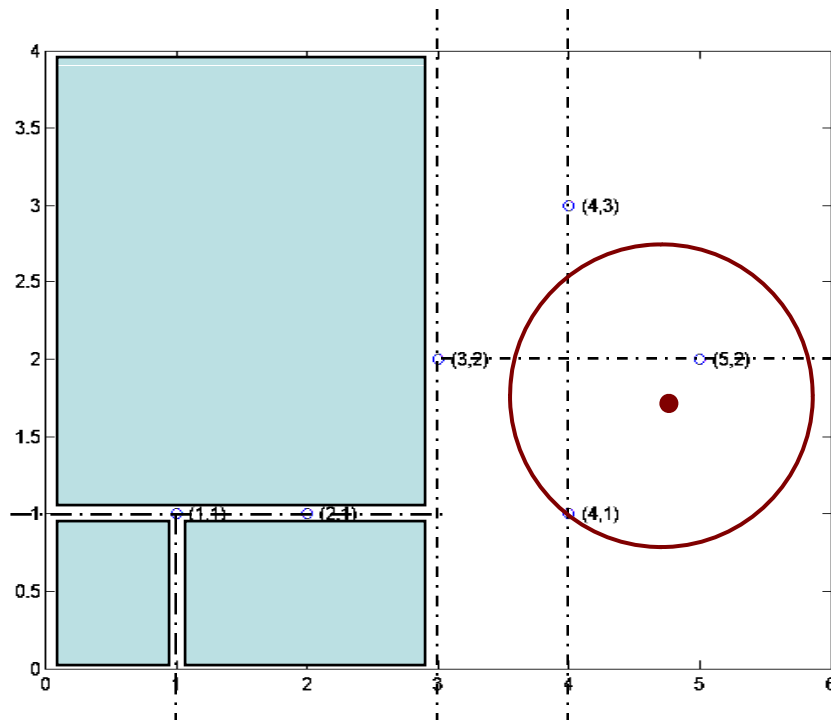
- We take any analytical solution to the pose problem
- As before, Umeyama method could be taken
- Other methods also possible
- This makes the ICP algorithm modular:  
Just use a suitable estimation algorithm and you can do 2D-2D, 2D-3D, 3D-3D registration depending on the input data

#### 4. Apply the motion $R, t$ to all points in $E_1$ and repeat until convergence

- When to converge?
- As in many iterative optimization schemes, the difference in cost function evaluations is deciding on termination
- $t_{old}$  and  $t_{new}$  are translations from previous and current iteration.
- $\alpha_{old}$  and  $\alpha_{new}$  are axes of rotation from previous and current iteration.
- If  $\|t_{old} - t_{new}\| < \epsilon_t$  &&  $\|\alpha_{old} - \alpha_{new}\| < \epsilon_\alpha$  terminate.

## Practical Considerations: KD-Trees for Efficient Point Set Structuring

- Successively separate the space in each dimension



→ Querying for closest point  
becomes a local problem!

## Practical Considerations: Distance Transforms

- Use distance map for getting closest points
- Computing closest points becomes a problem of map-lookup (interpolation)
- Sampling problem now correlates with the grid sampling in the distance map

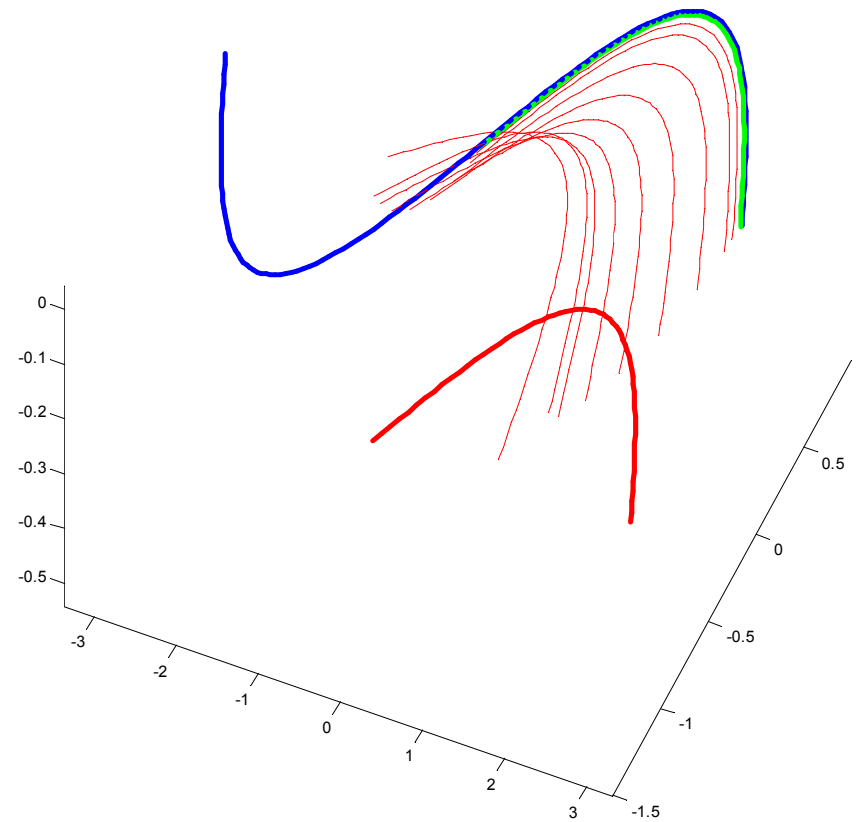
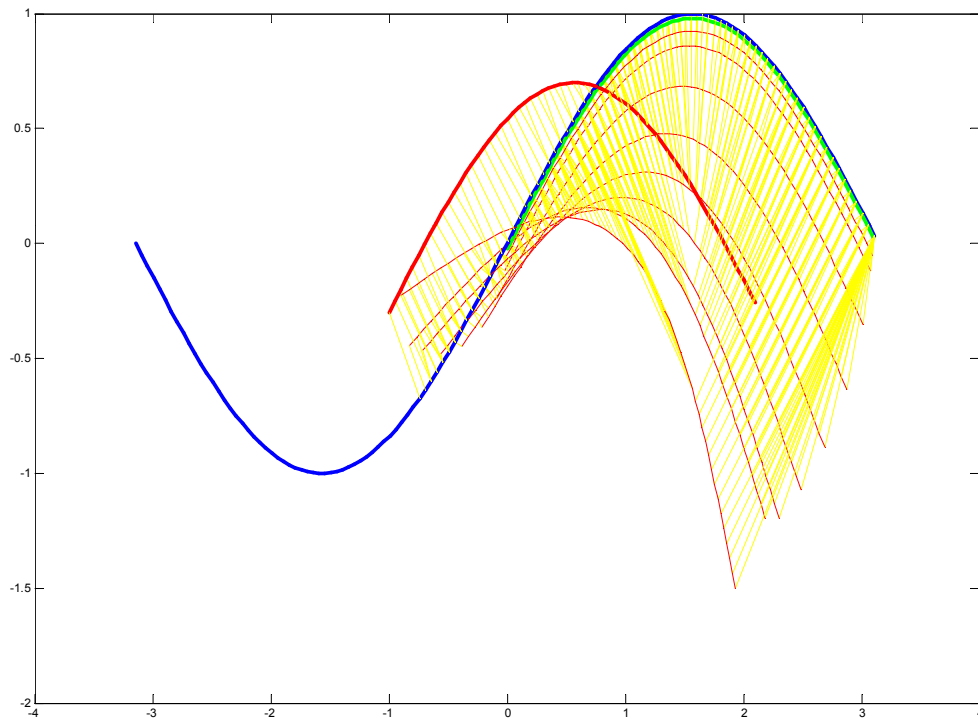
[curves]



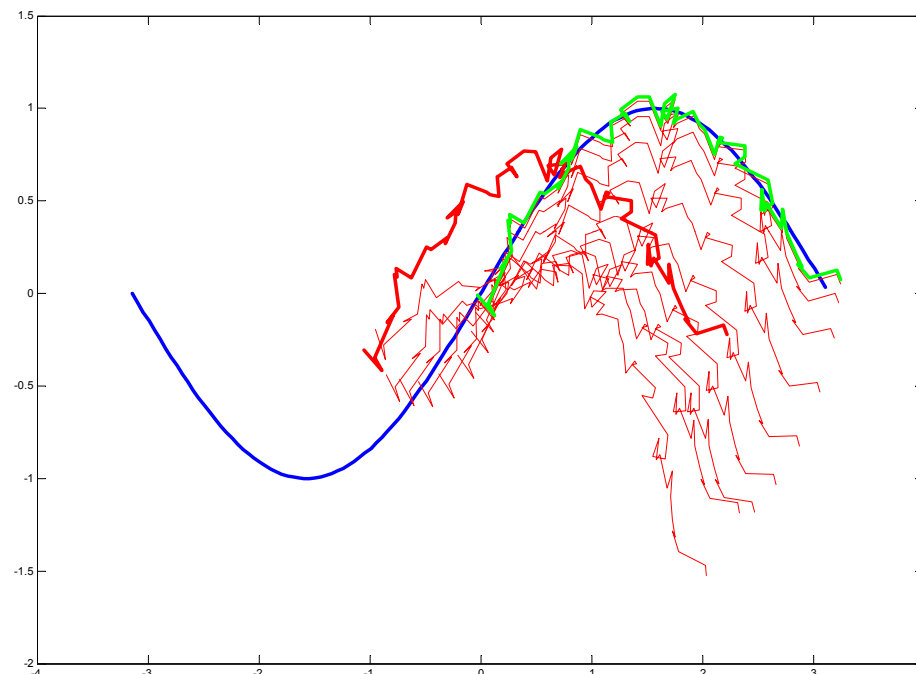
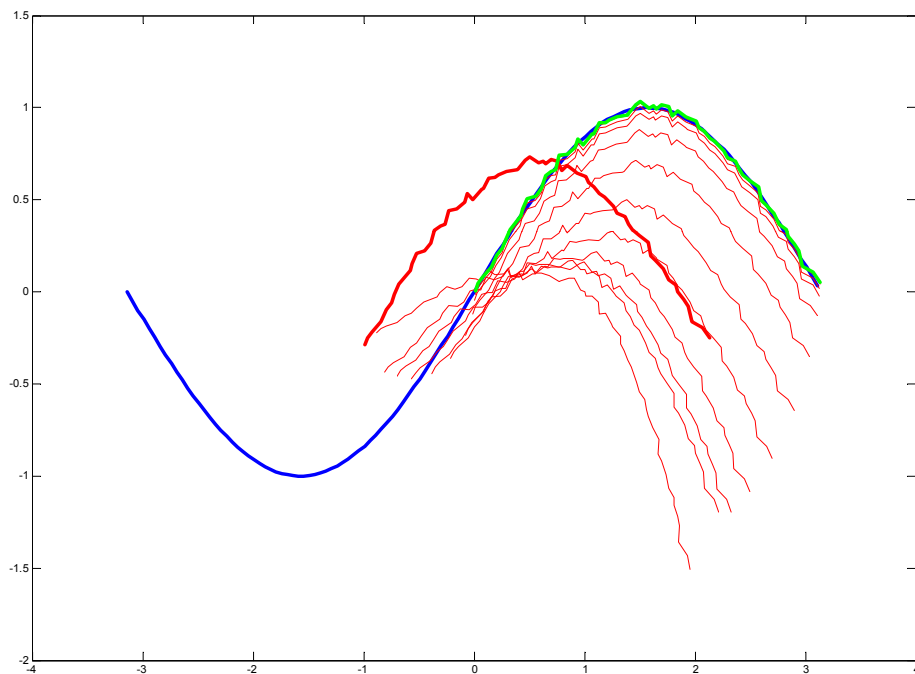
[Distance transform of curves]



# Iterative Closest Point: Example

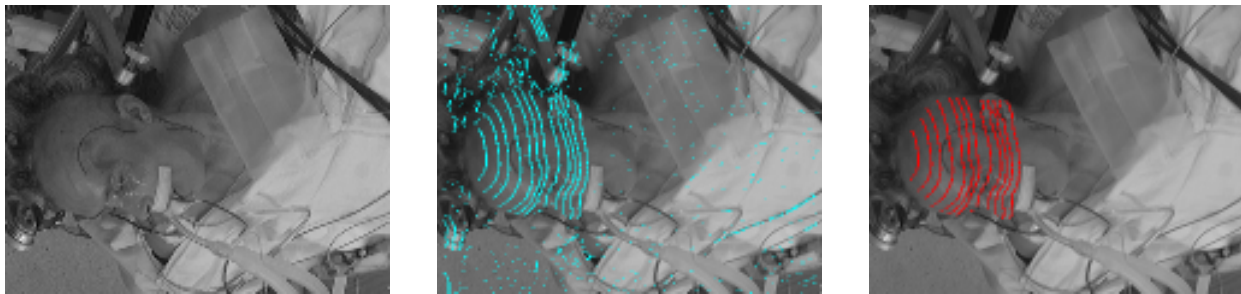


## Iterative Closest Point: Robustness



## Application: Intra-operative navigation

- 3D surface of the patient's skin is obtained with laser scanner



- ICP aligns it with the skin surface from a preoperative data set, e.g. Magnetic Resonance (MR) or Computed Tomography (CT).

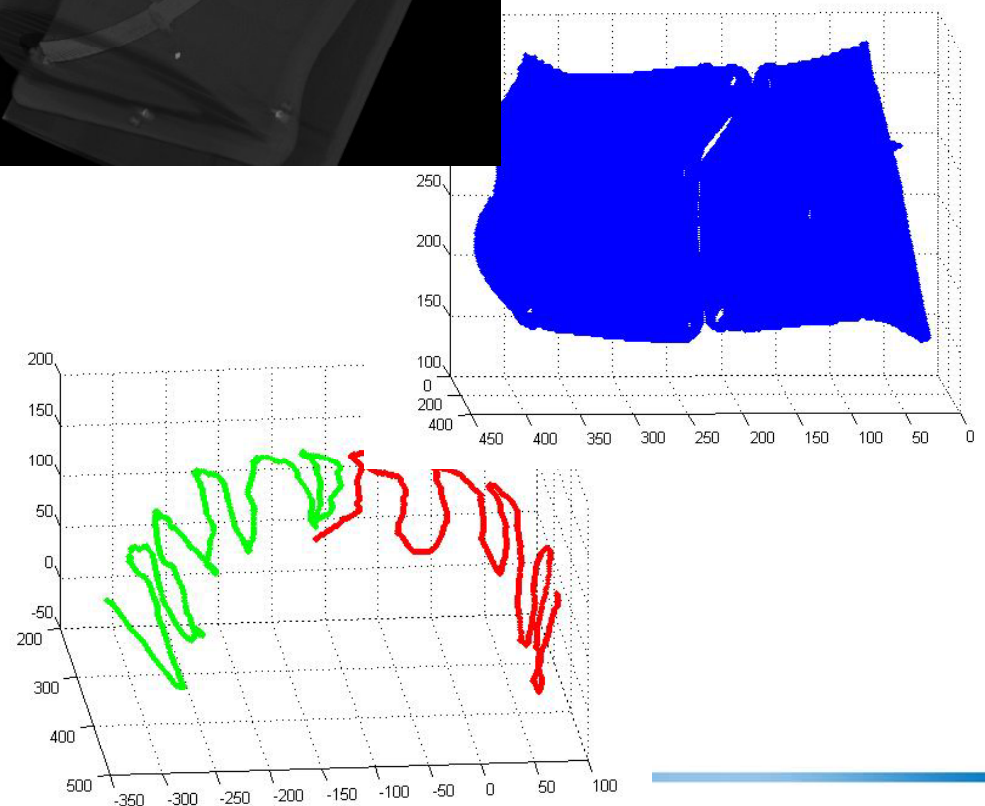
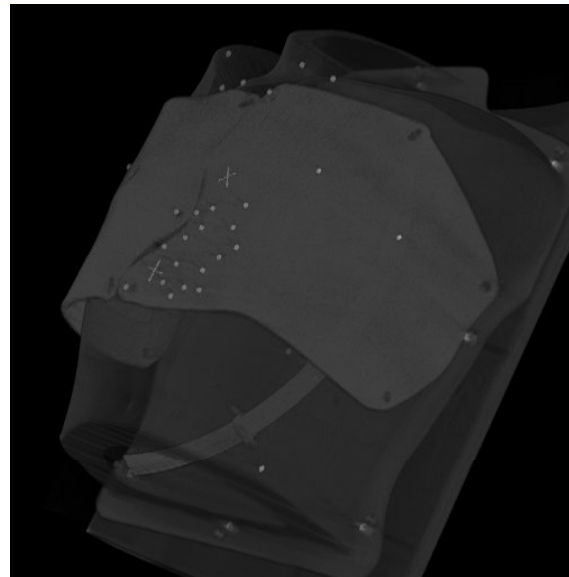


## References

- [1] Hajnal, Hill, Hawkes – *Medical Image Registration*
  - [2] Maintz, Viergever - *An Overview of Medical Image Registration Methods*
  - [3] Z. Zhang - *Iterative Point Matching for registration of free form curves*, March 1992
  - [4] Paul J. Besl and Neil D. McKay - *A Method for Registration of 3-D Shapes*, 1992
  - [5] A. Moore - *An introductory tutorial on kd-trees*, Technical Report No. 209, Computer Laboratory, University of Cambridge, 1991.  
[http://www.ri.cmu.edu/pubs/pub\\_2818.html](http://www.ri.cmu.edu/pubs/pub_2818.html)
  - [6] Lindsay I. Smith, *A Tutorial on Principal Components Analysis*, Feb 2002
  
  - Some slides are from presentations by Hesam Najafi.
-

## Assignment (part I)

- CT scan of phantom data
- Extracted surface represented as point cloud
- Tracking data from surface
- Match surface and tracking data

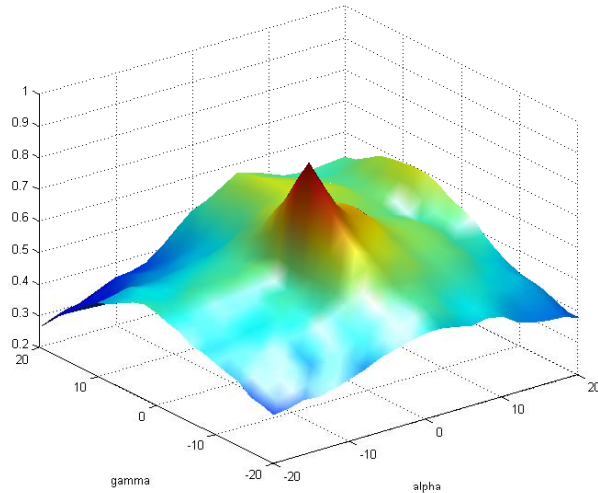




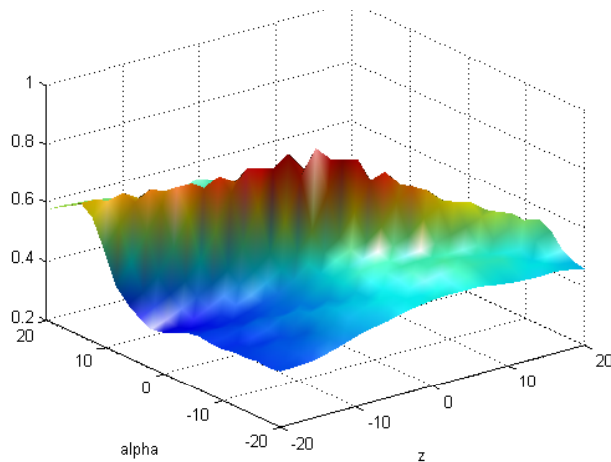
## Outline - Optimization

- Non-linear Optimization
- Direct Optimization
  - Best Neighbor
  - Powell-Brent
  - Downhill Simplex
- Gradient-based Optimization

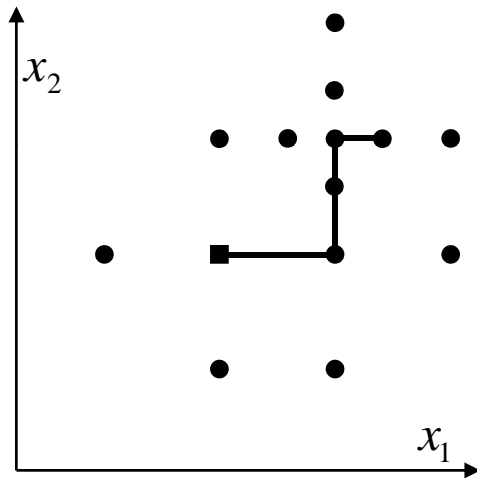
## Non-linear Optimization



- Function  $f: \mathbb{R}^n \rightarrow \mathbb{R}$  to be minimized
- Direct search: Simplex, Best Neighbor, Powell-Brent
- Gradient information available: Gradient Descent, (Gauss-) Newton



## Best Neighbor Optimizer

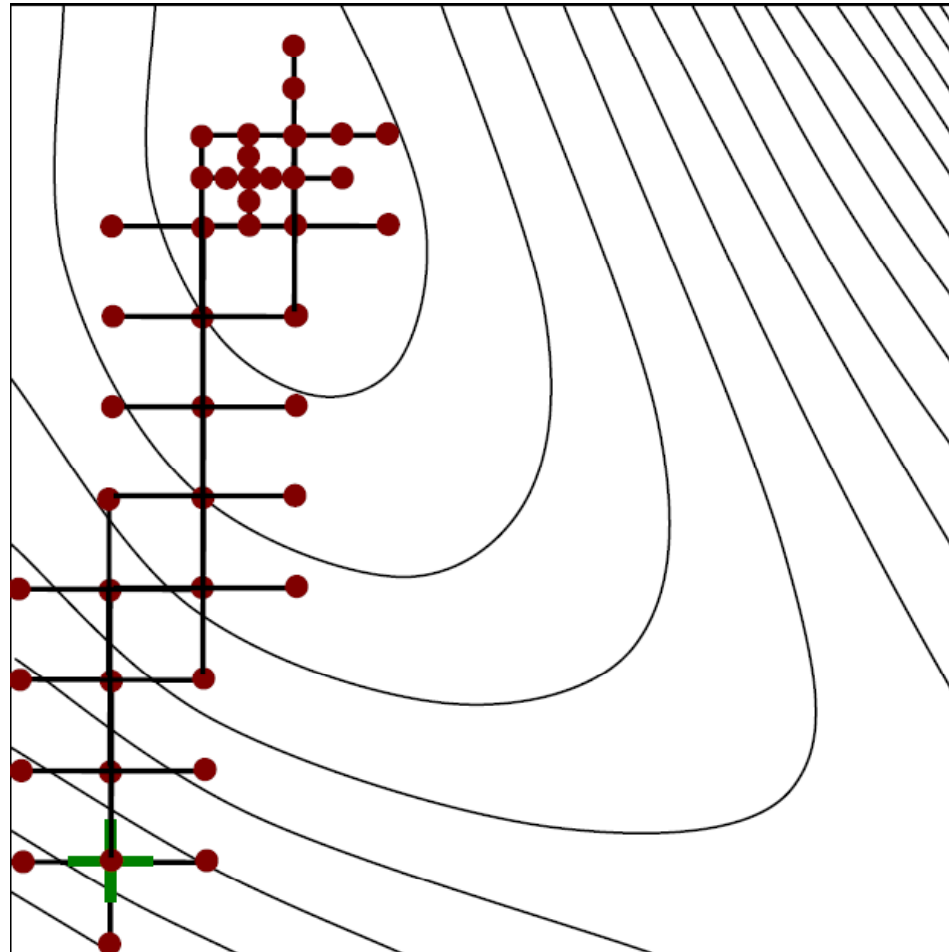


1. Evaluate all  $2n$  neighbors of the current location
2. If better than the current value, set best neighbor as next location, repeat 1.
3. Otherwise half the step size and try again
4. Abort if stepsize or cost function difference below limit

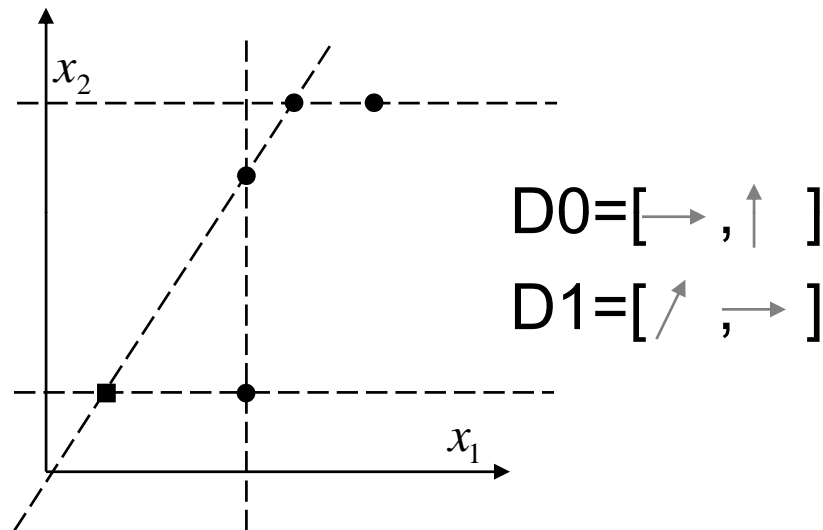
Works very stable – only locations are chosen that have been evaluated

## Best Neighbor Optimizer (2)

Applied to two parameters of a registration problem



## Powell-Brent Optimizer



### Each Iteration:

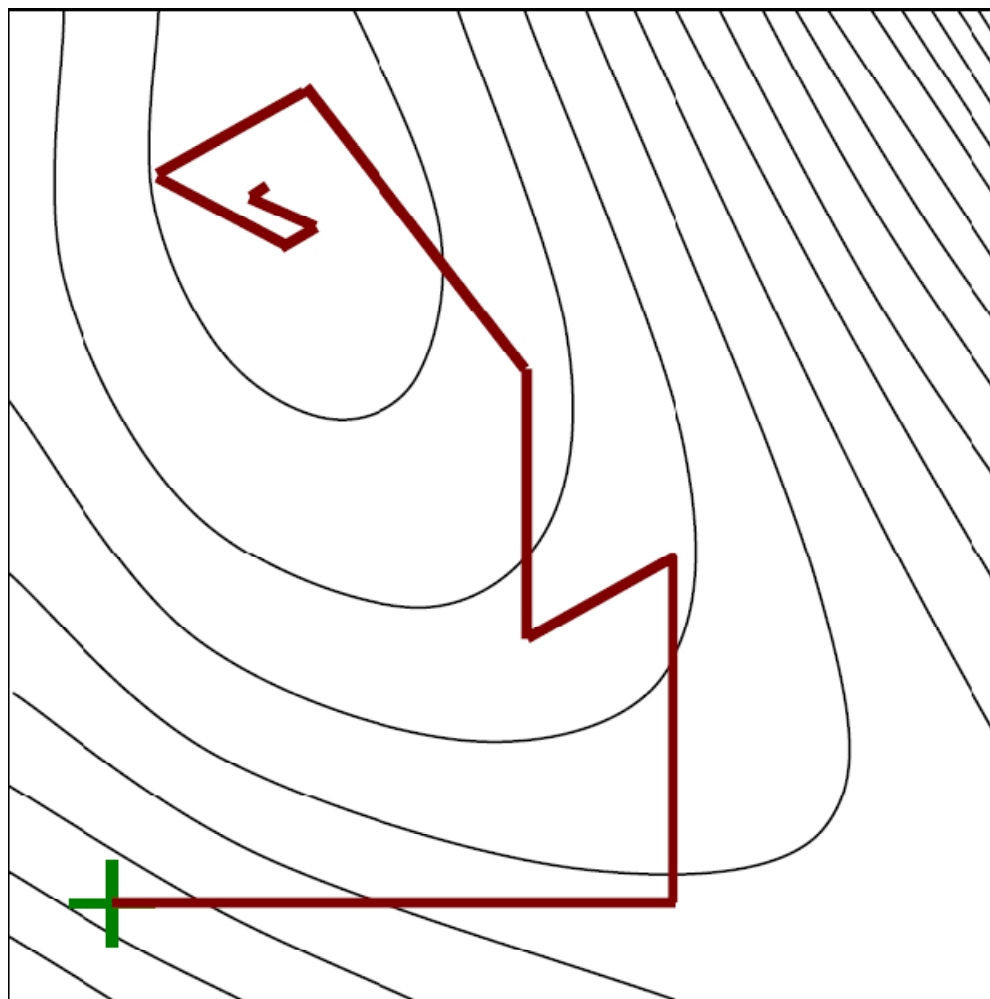
- n successive line minimizations
- Adaptation of the direction set

### Advantages:

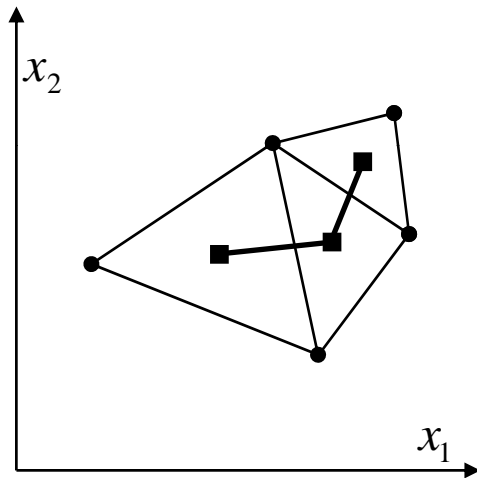
- No gradient information needed
- Few cost function evaluations

## Powell-Brent Optimizer (2)

Applied to two parameters of a registration problem



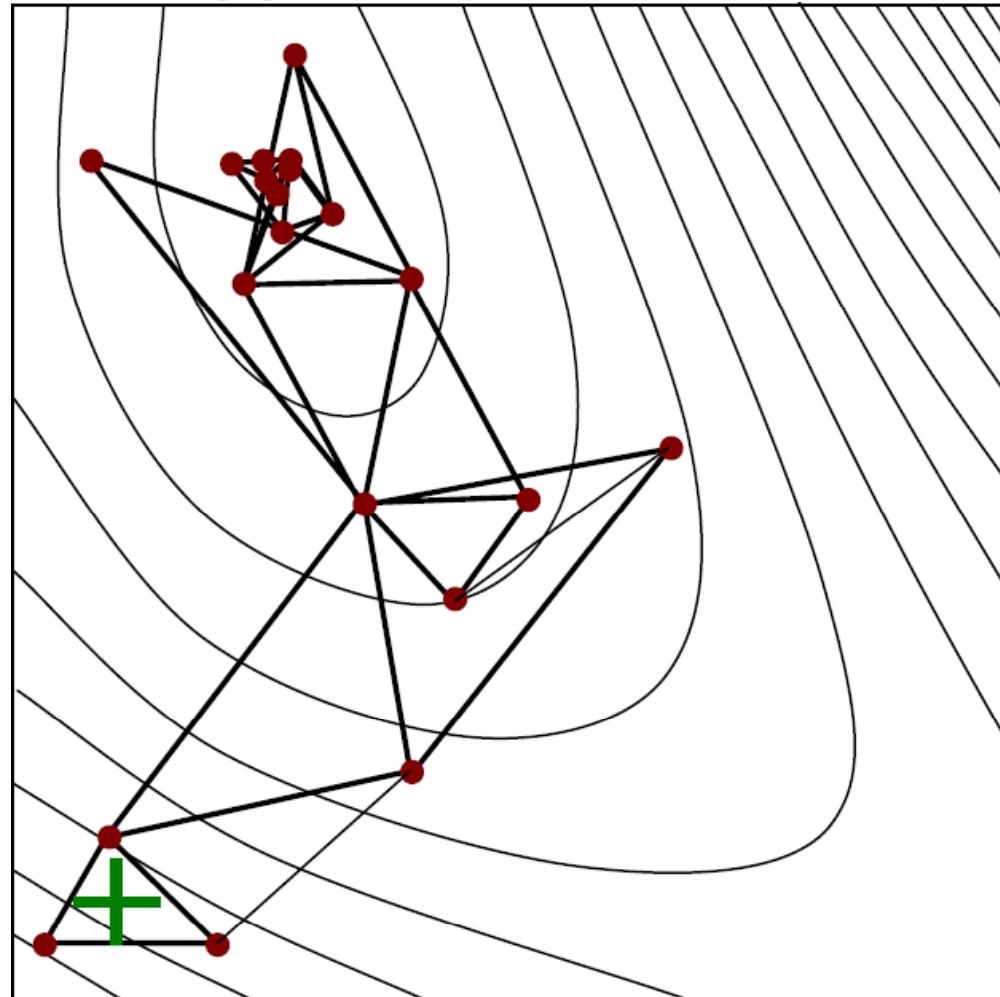
## Downhill-Simplex Optimizer



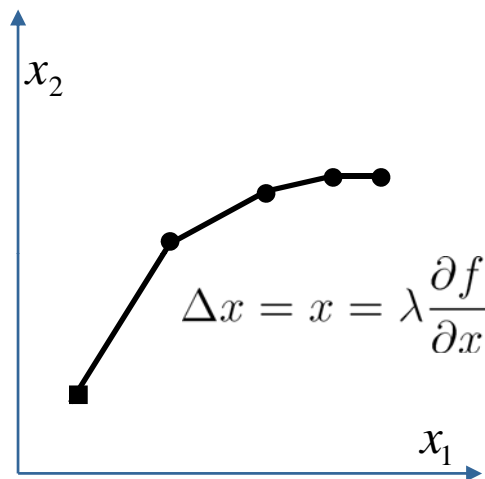
- Simplex: Minimal geometric shape in  $n$  dimensions,  $n+1$  vertices
- Position and shape of Simplex is altered during optimization
  - *Reflection*: move “highest point” to opposite side (preserves volume of simplex)
  - *Expansion*: if possible, expand simplex in a direction to enlarge the step size
  - In a “valley floor” *contract* simplex in transverse direction and move down the valley
  - “pass through the eye of a needle”, *contract* in each direction towards the “lowest point”
- Very few cost function evaluations
- Have a look at [2]

## Downhill-Simplex Optimizer (2)

Applied to two parameters of a registration problem



## Steepest Descent Optimizer



- Update equation for each optimizer

$$x_{new} \leftarrow x + \lambda \Delta x$$

→ why not use gradient in  $\Delta x$  computation?

- Steepest Descent - compute  $\Delta x$  from 1<sup>st</sup> derivative:

$$\Delta x = -\frac{\partial f}{\partial x}$$

- Gradient delivers direction of steepest ascent → fast convergence

- How to determine  $\lambda$ ? Line search, look for

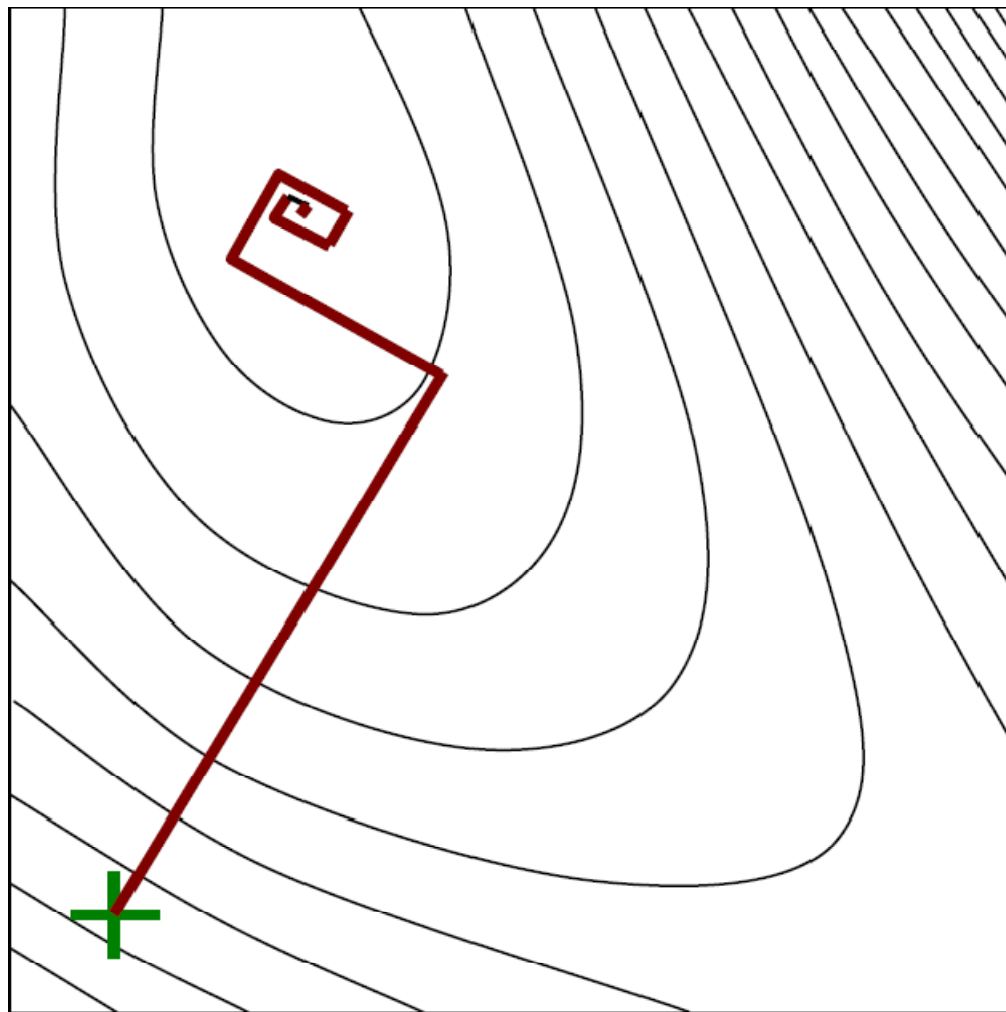
$$\arg \min_{\lambda} \varphi(\lambda), \text{ where } \varphi(\lambda) = x + \lambda \Delta x$$

- Steepest Descent Optimizer are slow due to line search for  $\lambda$  and gradient computation

- Even faster (quadratic) convergence when involving higher-order derivatives: Newton

## Gradient Descent Optimizer (2)

Applied to two parameters of a registration problem



## References

- **[1] W. Press et al.:** *Numerical Recipes in C*, Second Edition, CRC Press, Inc., 1992.
- **[2] Java Applet for Simplex Optimization:**  
[http://www.chem.uoa.gr/applets/AppletSimplex/AppI\\_Simplex2.html](http://www.chem.uoa.gr/applets/AppletSimplex/AppI_Simplex2.html)