

# Computer Aided Medical Procedures II

## Atlas-based Segmentation

Ben Glocker and Nassir Navab

10 May 2007

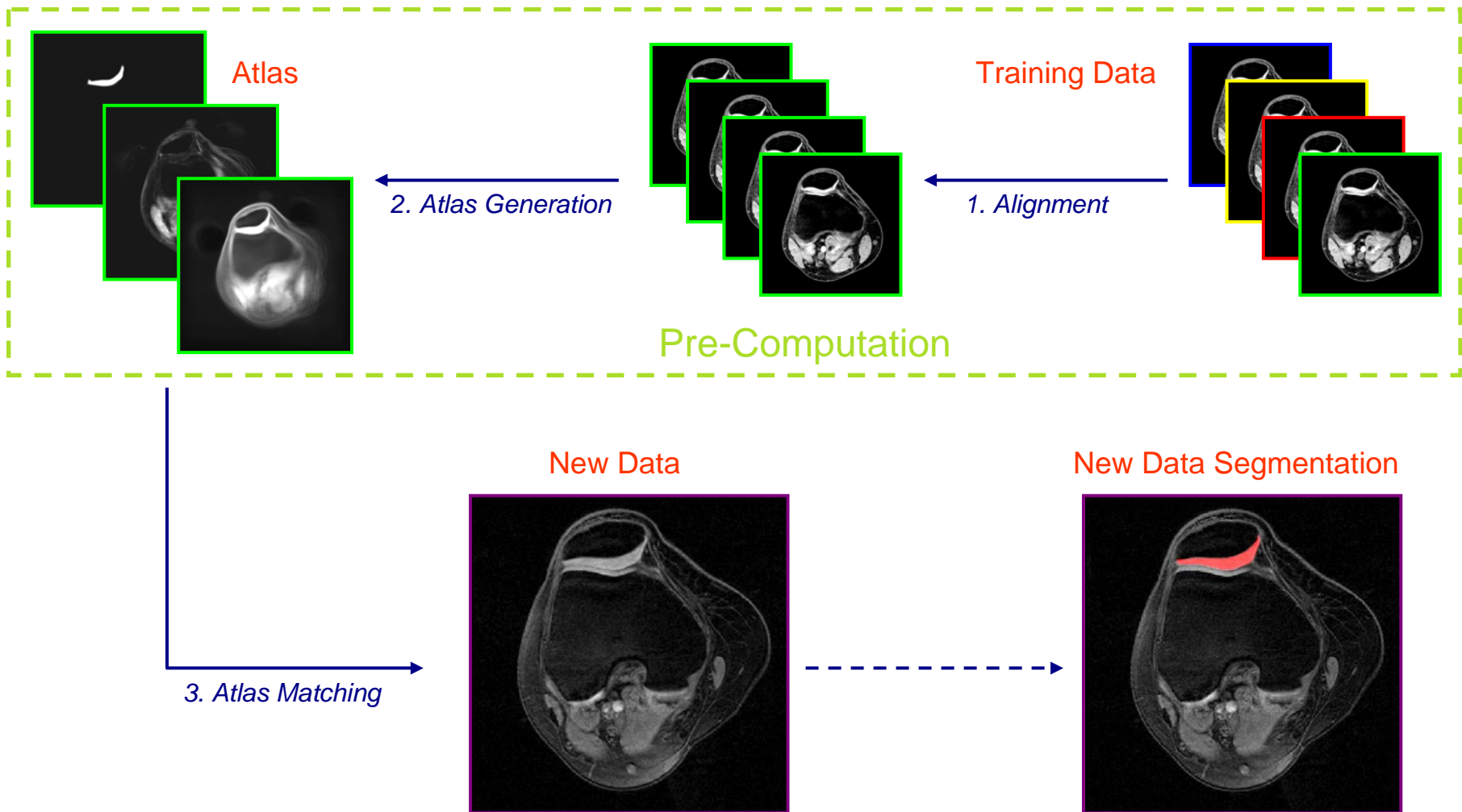
chair for computer aided medical procedures & augmented reality

department of computer science | technische universität münchen

# Prerequisites

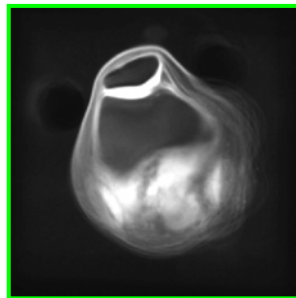
- Training Data
  - Set of segmented data of the same anatomy used for building an atlas
- Data Correspondence
  - Training data has to be aligned such that corresponding points can be identified
- Atlas Model
  - A (statistical) model that represents the atlas
- Atlas Matching Algorithm
  - Algorithm that is able to fit the atlas to a new data set

# Scheme for Atlas-based Segmentation

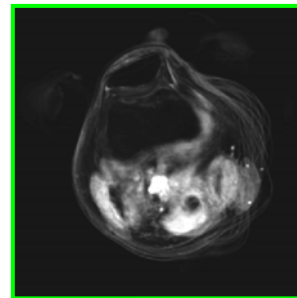


# Types of Atlases

- Image Appearance Models
  - Only the appearance (e.g. intensity) of training data is considered
  - Image deformation model is needed (e.g. Free Form Deformations)



Mean Image



Variance Image



Segmentation

# Types of Atlases (continued)

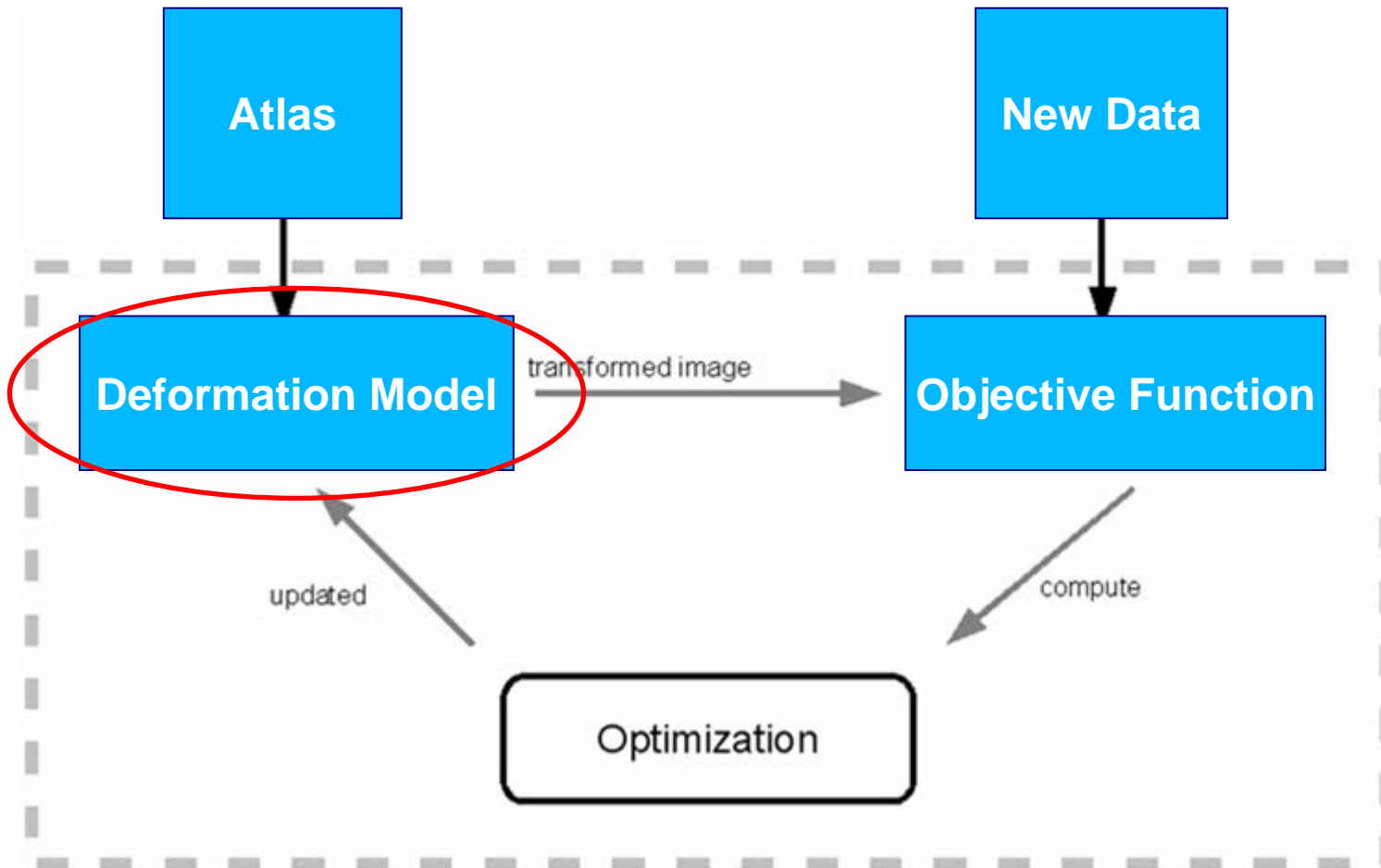
- Shape Models
  - Shape of the segmented training data is considered (surface extraction)
  - Mean shape and variations are computed via Principal Component Analysis
  - At each sample point a feature vector is stored (e.g. intensity, gradient, texture, curvature)



*Images by Juergen Fripp et al.*

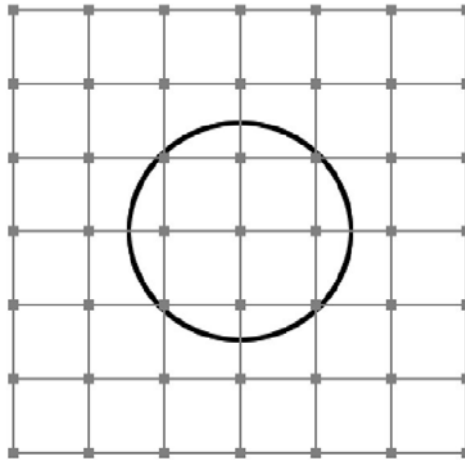
# Atlas Matching for Image Appearance Models

# Scheme for Atlas Matching



# Image Deformation Model

- Free Form Deformations
- Regular grid with uniformly spaced control points  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
- Additionally, *virtual control points* are used for border handling



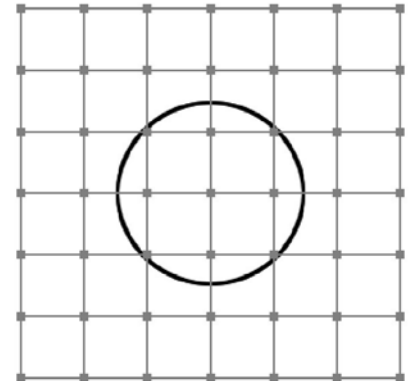
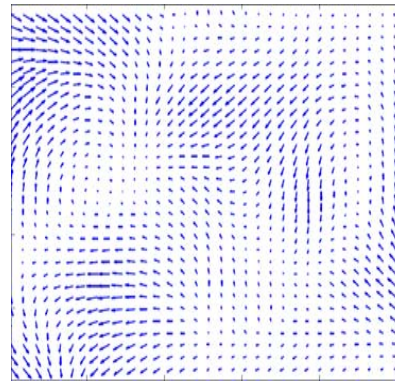
# Free Form Deformations (FFD)

- Transformation (3D) based on cubic B-Splines

$$\mathbf{D}(\mathbf{x}) = \sum_{l=0}^3 \sum_{m=0}^3 \sum_{n=0}^3 B_l(u) B_m(v) B_n(w) \mathbf{d}_{i+l, j+m, k+n}$$

- with polynomials

$$\begin{aligned} B_0(u) &= (-u^3 + 3u^2 - 3u + 1)/6 \\ B_1(u) &= (3u^3 - 6u^2 + 4)/6 \\ B_2(u) &= (-3u^3 + 3u^2 + 3u + 1)/6 \\ B_3(u) &= u^3/6. \end{aligned}$$



- and  $u$ ,  $v$ , and  $w$  are the relative positions of point  $\mathbf{x} = (x, y, z)^T$

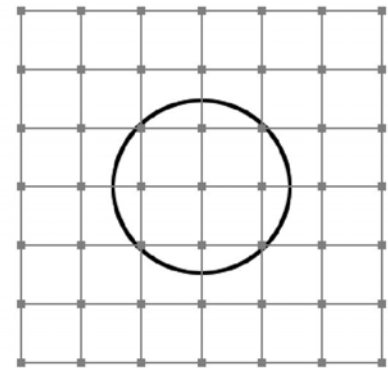
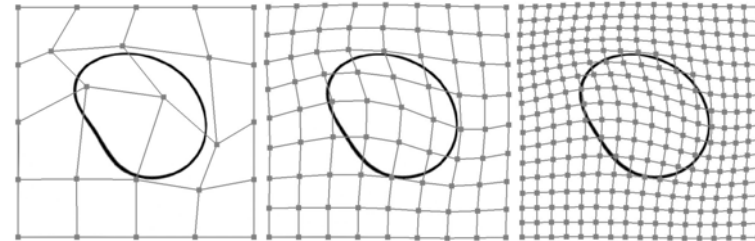
# Properties of FFD

## Advantages

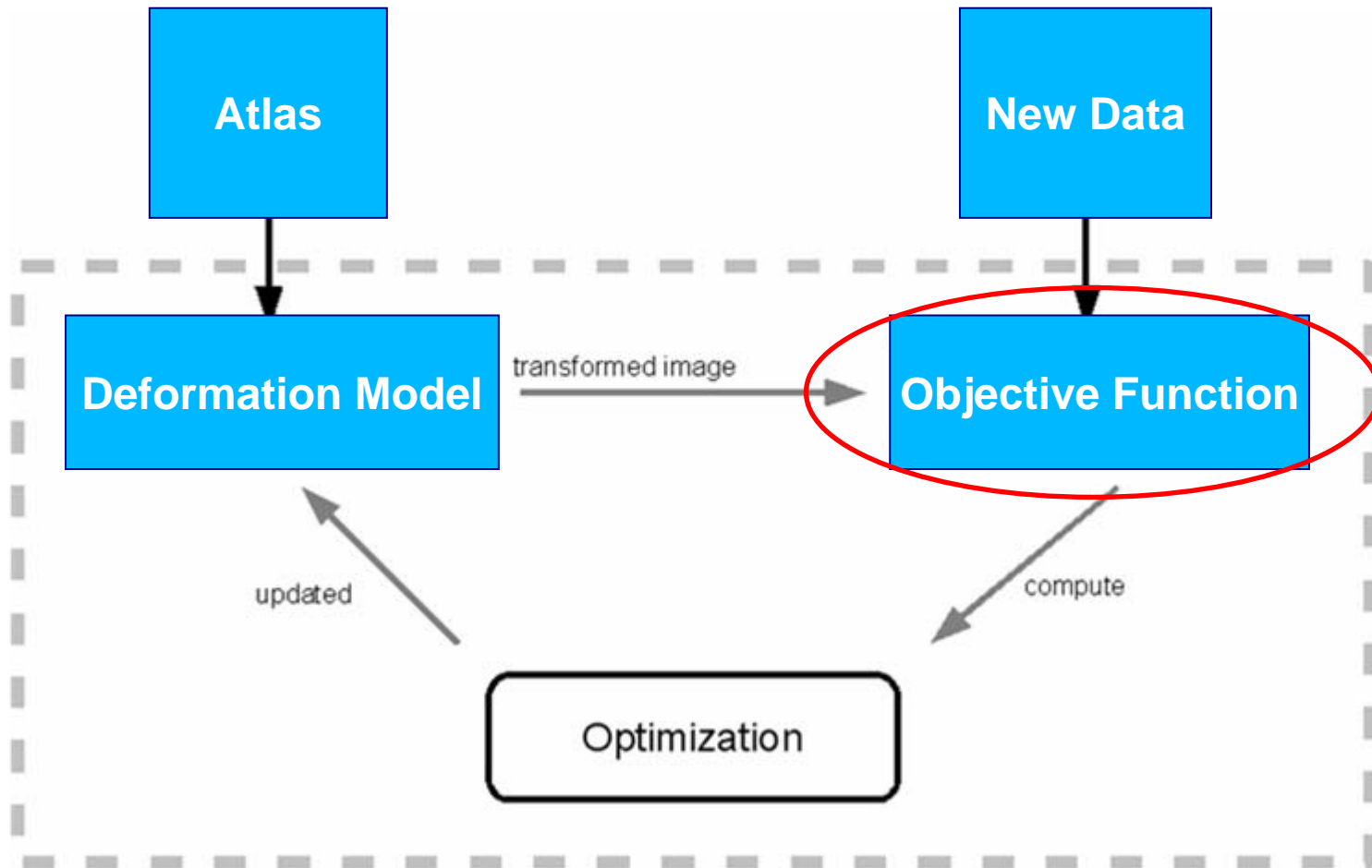
- Local influence of each control point
- $C^2$  continuous transformations
  - *Smooth transitions*
- Multi-level approach allows for large and small deformations simultaneously

## Disadvantages

- In general, there is no *one-to-one* correspondence between control grid movements and image deformation
- Inverse transformation is not trivial to determine



# Scheme for Atlas Matching



# Objective Function

- Objective function is a quantitative measure how well the current **parameters** fit to the **input data**
- It consists of two parts
  - The matching criteria
  - A smoothness penalty

$$E : (l, d) \mapsto R$$

Scalar-valued function

$$E(l, d) = \mathcal{F} + \lambda \mathcal{S}$$

Matching criteria      Weighting      Smoothness penalty

The diagram illustrates the structure of the objective function. At the top, the function is defined as  $E : (l, d) \mapsto R$ , with a dashed orange arrow pointing from the text 'Scalar-valued function' to it. Below this, the function is expanded as  $E(l, d) = \mathcal{F} + \lambda \mathcal{S}$ . Three dashed orange arrows point from the terms below to the components of the equation: 'Matching criteria' points to  $\mathcal{F}$ , 'Weighting' points to  $\lambda$ , and 'Smoothness penalty' points to  $\mathcal{S}$ .

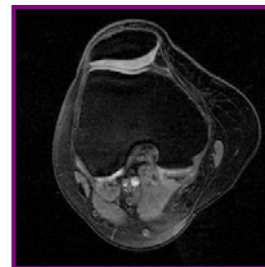
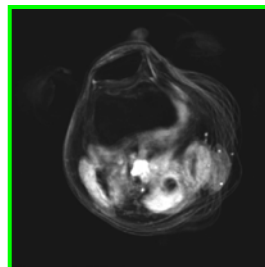
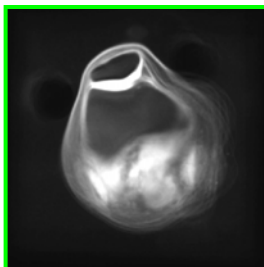
# Atlas Matching Criteria

- Naive
  - Sum of Squared Differences

$$\mathcal{F}_{SSD}(I_{\mu}, I_{\text{new}}) = \frac{1}{|\Omega|} \sum_{\mathbf{x} \in \Omega} (I_{\mu}(\mathbf{x}) - I_{\text{new}}(\mathbf{x}))^2$$

- Better: Statistical Measure

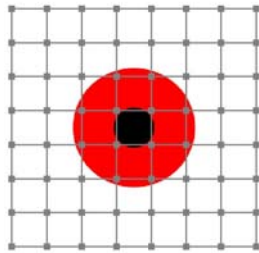
$$\mathcal{F}_{Atlas}(I_{\mu}, I_{\sigma^2}, I_{\text{new}}) = \frac{1}{|\Omega|} \sum_{\mathbf{x} \in \Omega} \frac{(I_{\mu}(\mathbf{x}) - I_{\text{new}}(\mathbf{x}))^2}{2I_{\sigma^2}(\mathbf{x})}$$



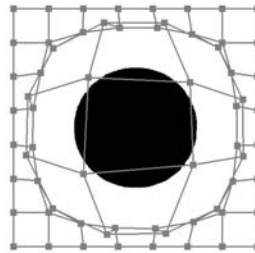
# Smoothness Penalty – Why?

- Many optimization problems are ill-posed
- Often presence of noise or corrupted data
- Smoothness penalty avoids undesired solutions
  - Example

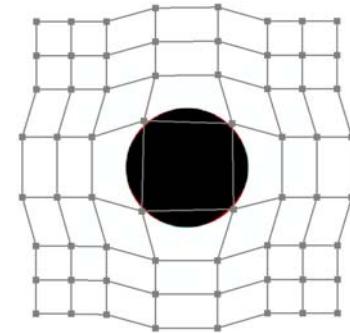
$$\mathcal{S} = \frac{1}{|\mathcal{E}|} \sum_{(p,q) \in \mathcal{E}} |\mathbf{d}_p - \mathbf{d}_q|$$



Initial Grid

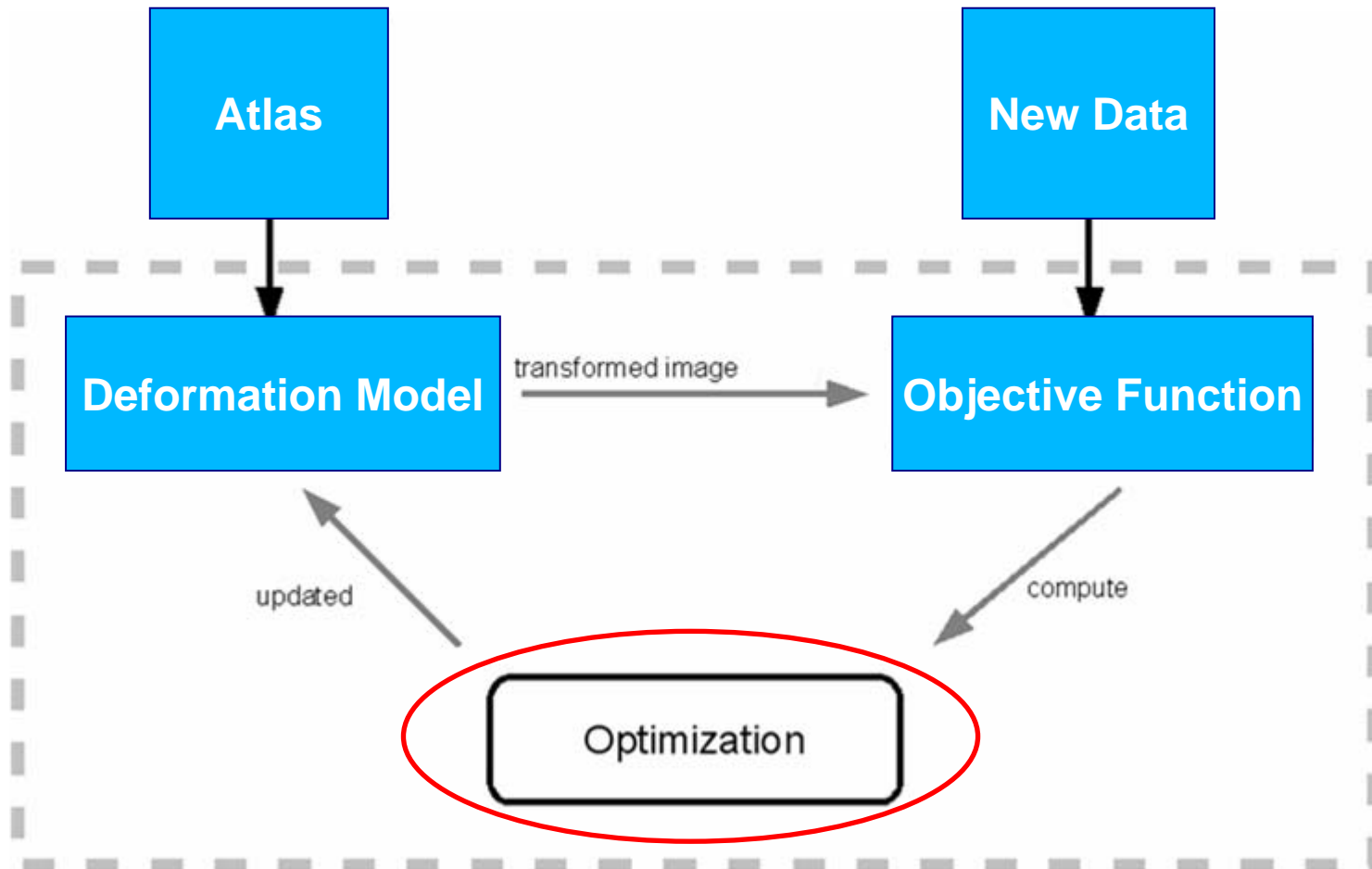


No Smoothness Penalty



With Smoothness Penalty

# Scheme for Atlas Matching



# Optimization

- According to the objective function the optimization algorithm tries to find the optimal parameters
- **In our case:** parameters are the control point positions of the deformation model
- Numerous optimization algorithms

$$l_{opt} = \arg \min_l E(l, d)$$

$l$  : set of parameters

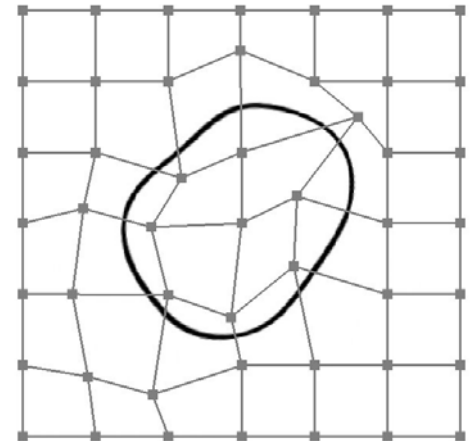
$d$  : input data (atlas + new data)

$E$  : objective function (including matching criteria)

# Discrete Optimization

**Optimization problem**  $l_{opt} = \arg \min_l E(l, d)$

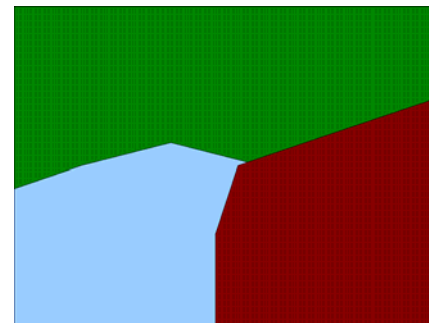
- A set of *discrete variables*  $\mathcal{V}$  models the problem parameters
  - *i.e. control points of FFD model*
- A set of *labels*  $\mathcal{L}$  represents the hidden information
  - *i.e. displacements of the control points*
- An estimation  $l$  is a *discrete labeling*  $l : \mathcal{V} \mapsto \mathcal{L}$ 
  - *i.e. specific grid configuration*



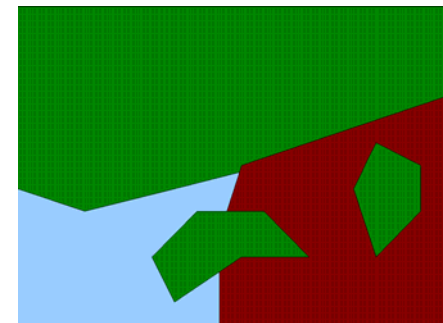
# Alpha-Expansion Algorithm

1. Start with an arbitrary labeling  $l$
2. Set `success := 0`
3. For each label  $\alpha \in \mathcal{L}$ 
  - 3.1 Find  $\hat{l} = \arg \min E(l, d)$  among  $l'$  within one  $\alpha$ -expansion of  $l$
  - 3.2 If  $E(\hat{l}, d) < E(l, d)$ , set  $l := \hat{l}$  and `success := 1`
4. If `success = 1` goto 2
5. Return  $l$

Binary Problem



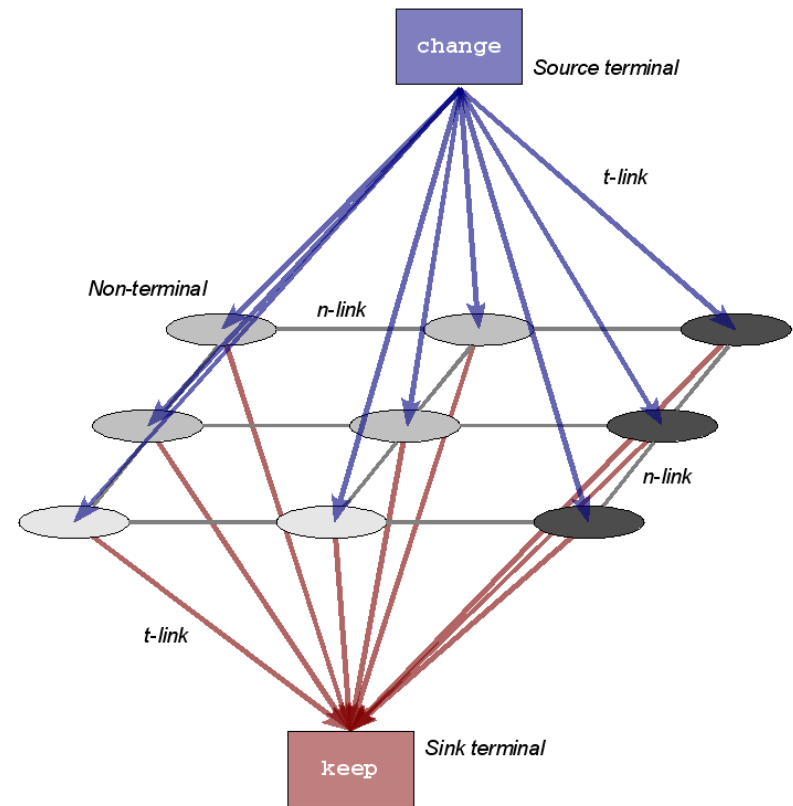
Labeling



Green-Expansion

# Optimal Expansion Move via Graph Cuts

- Graph cuts represent all possible expansion moves
  - **Non-terminals** represents the set of variables  $\mathcal{V}$
  - **Source** terminal represents the expanded label  $\alpha \in \mathcal{L}$
  - **Sink** terminal represents the current position
  - **Min-cut/max-flow** solution gives the optimal expansion
- **Weights of t-links and n-links?**





# Segmentation Mapping

- Apply the determined deformation field

