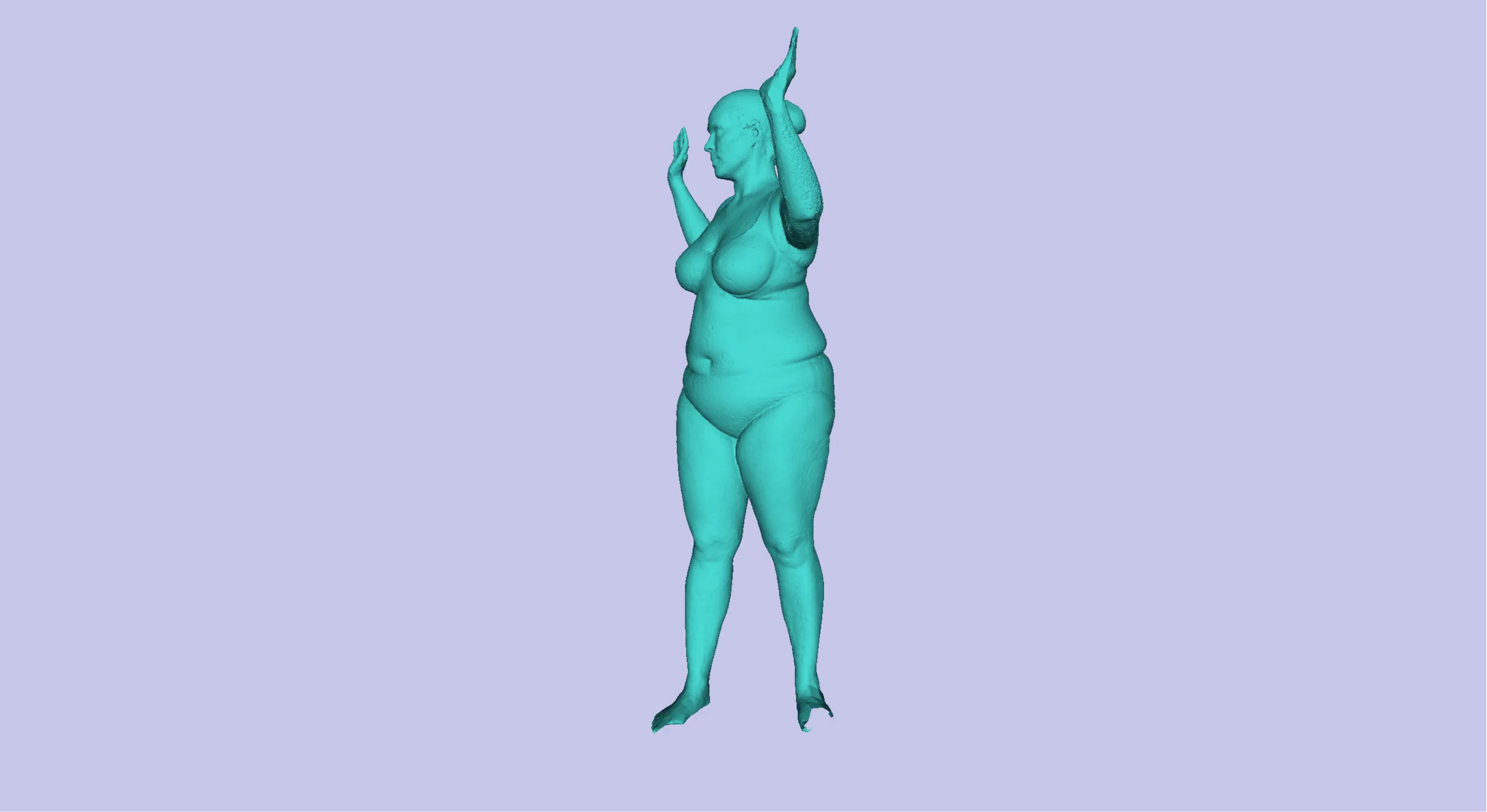
Virtual Humans – Winter 23/24

Lecture 4_1 – ICP: Iterative Closest Points

Prof. Dr.-Ing. Gerard Pons-Moll
University of Tübingen / MPI-Informatics

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Non-rigid Articulated Registration



What is missing?

Given correspondences, we can find the optimal rigid alignment with Procrustes.

PROBLEMS:

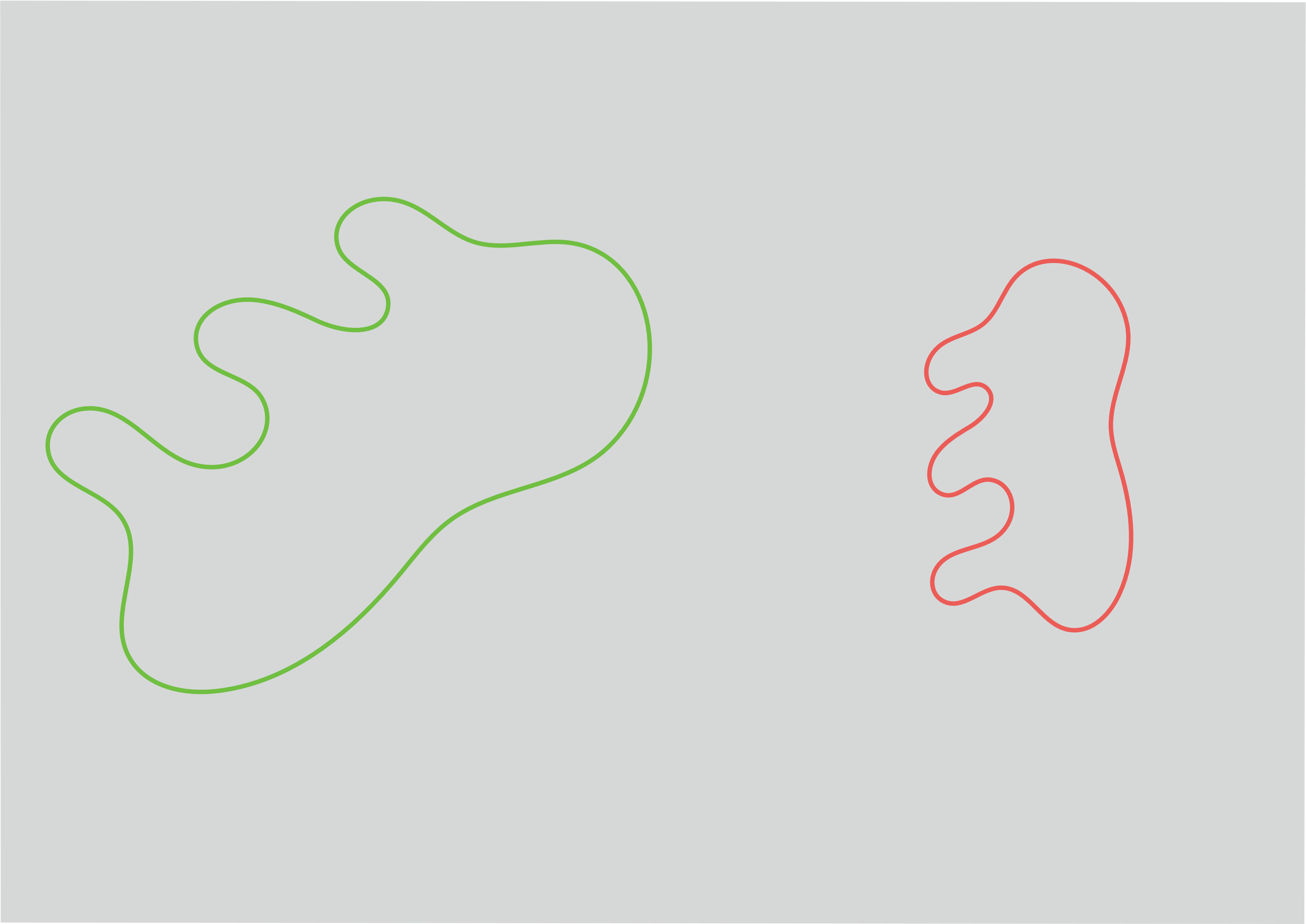
- How do we find the **correspondences** between shapes ?
- How do we **align** shapes **non-rigidly** ?

ICP and alignment based on optimisation

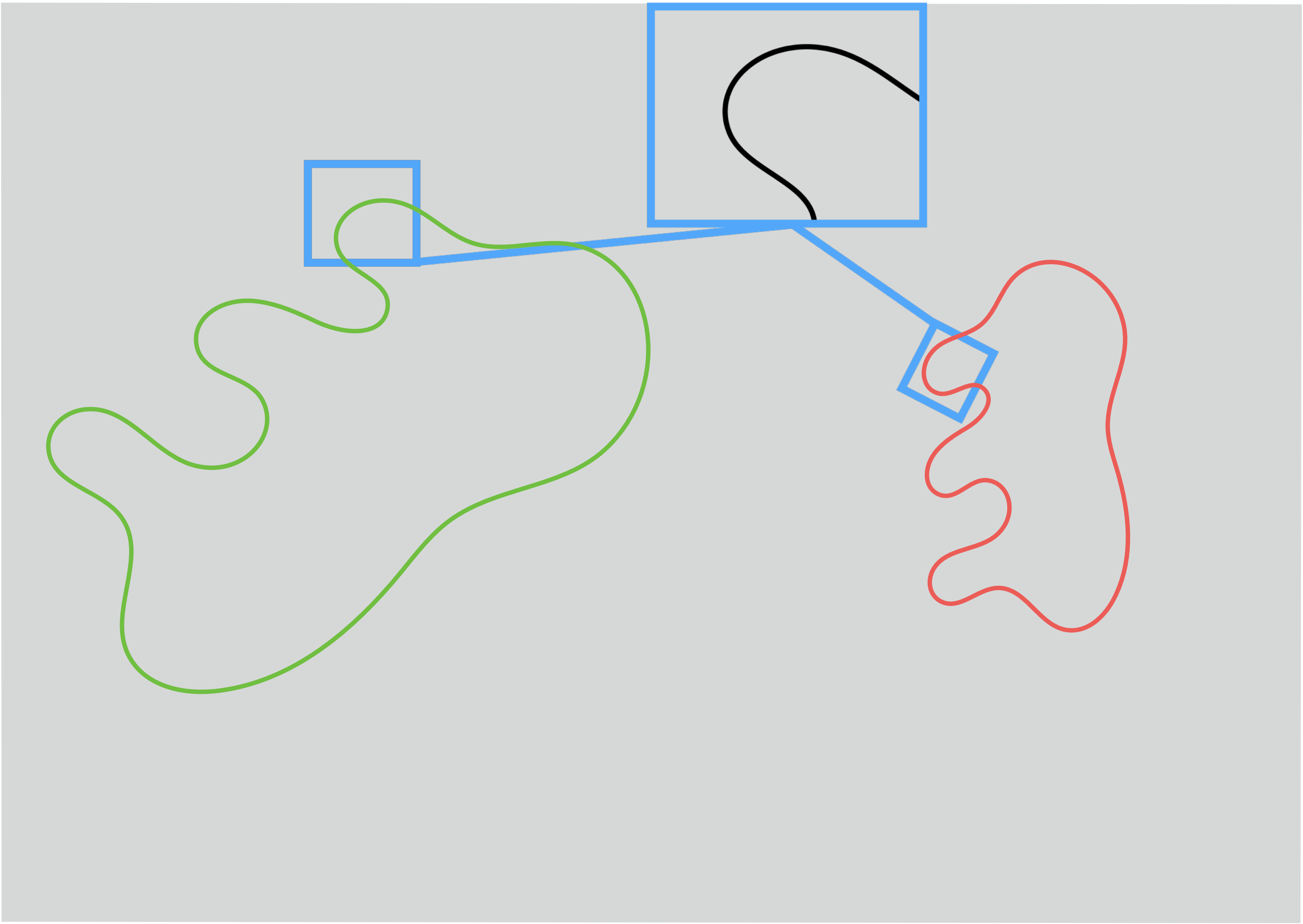
- Optimising alignment and correspondences using *Iterative Closest Point (ICP).*

- Alignment through *continuous* optimisation.

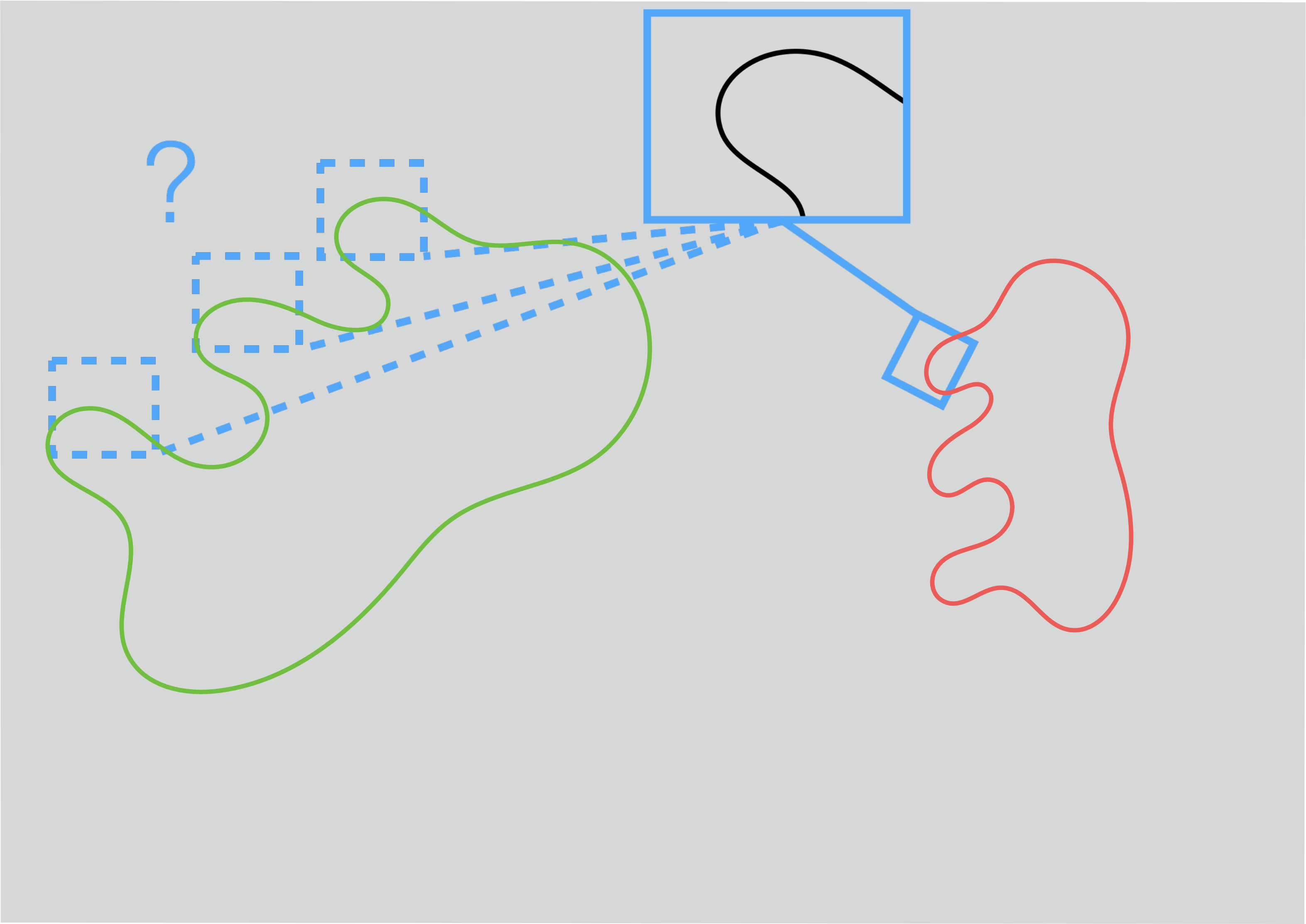
How do we find correspondences?



How do we find correspondences?



How do we find correspondences?



How do we find **correspondences**?

$$E \equiv \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2 \equiv \sum_i \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2$$

?

compact notation: f contains translation, rotation and isotropic scale

$\mathbf{x}_i$ Closest point to target shape point $\mathbf{y}_i$

The optimisation is over:

- the transform f
- the correspondences $\mathcal{C} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_i^N$

$$E(\mathcal{C}, f) = \sum_i \min_{\mathbf{x} \in \mathbf{X}} \|f(\mathbf{x}) - \mathbf{y}_i\|^2$$

How do we find correspondences?

The idea was to minimise the sum of distances between the one set of points and the other set, transformed

$$E \equiv \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2 \equiv \sum_i \|f(\mathbf{x}_i) \underset{?}{-} \mathbf{y}_i\|^2$$

compact notation: f contains translation, rotation and isotropic scale

Ideas

The idea was to minimise the sum of distances between the one set of points and the other set, transformed

$$E \equiv \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2 \equiv \sum_i \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2$$

compact notation: f contains translation, rotation and isotropic scale

What if we estimate the correspondences?

Solution: Iteratively find correspondences

The idea was to minimise the sum of distances between the one set of points and the other set, transformed

$$E \equiv \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2 \equiv \sum_i \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2$$

compact notation: f contains translation, rotation and isotropic scale

What if we estimate the correspondences?

iteration

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

original unsorted points

$$f^{j+1} = \arg\min_f \sum_i \|f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i\|^2$$

Alternate between finding correspondences and finding the optimal transformation

The idea was to minimise the sum of distances between the one set of points and the other set, transformed

$$E \equiv \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2 \equiv \sum_i \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2$$

compact notation: f contains translation, rotation and isotropic scale

What if we estimate the correspondences?

iteration

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

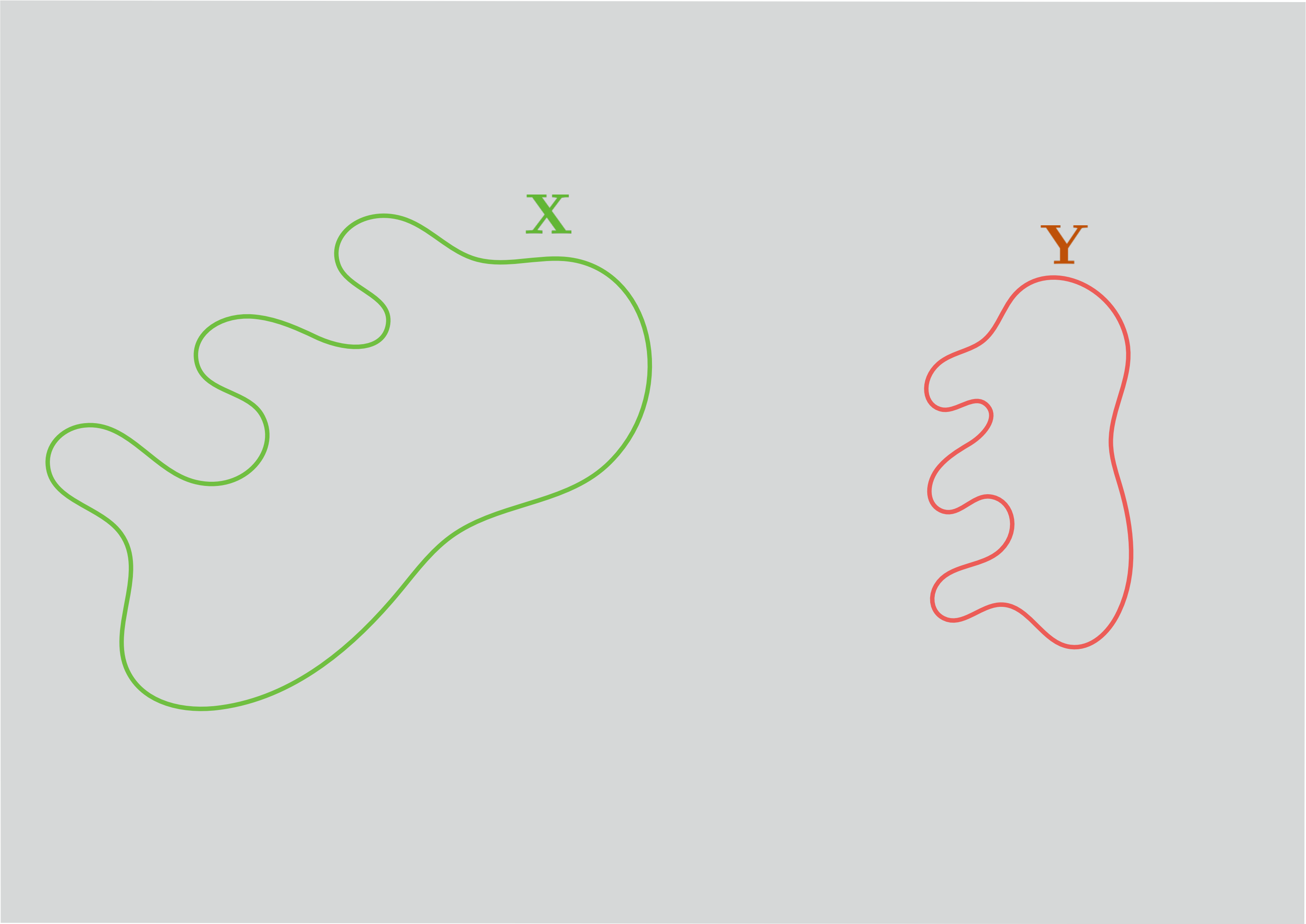
Given current best transformation, which are the closest correspondences?

original unsorted points

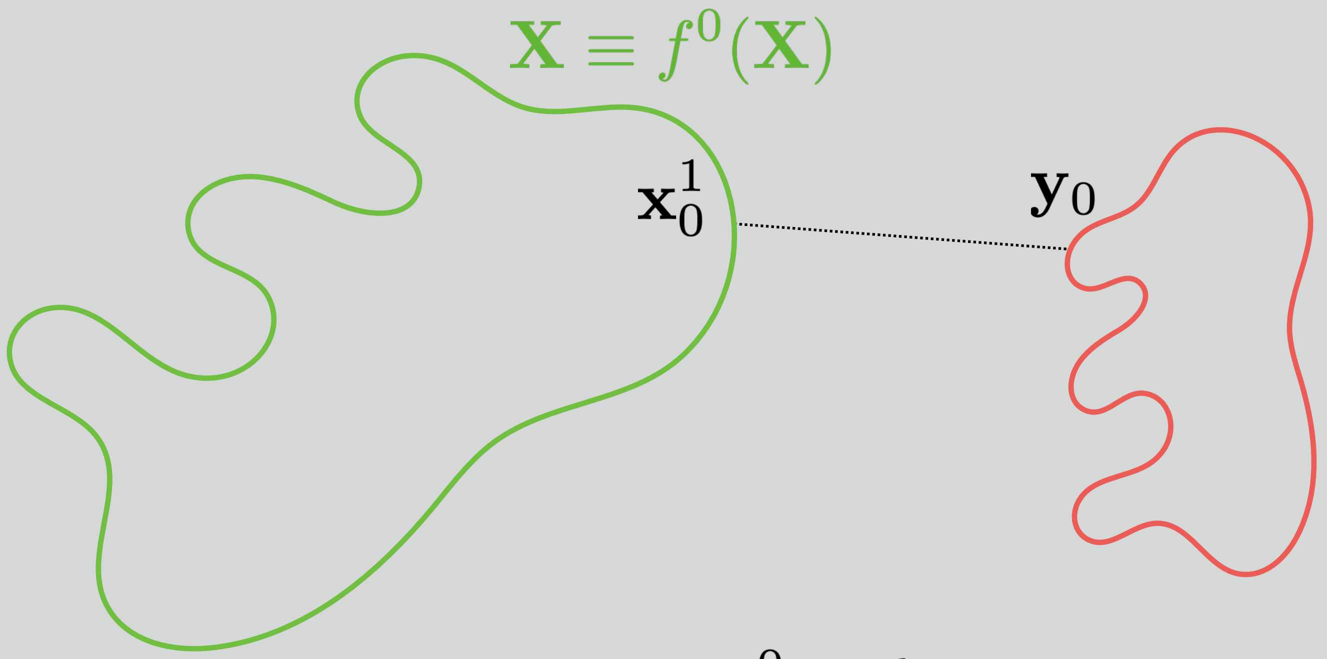
$$f^{j+1} = \arg\min_f \sum_i \|f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i\|^2$$

Given current best correspondences, which is the best transformation?

Make up reasonable correspondences



Make up reasonable correspondences

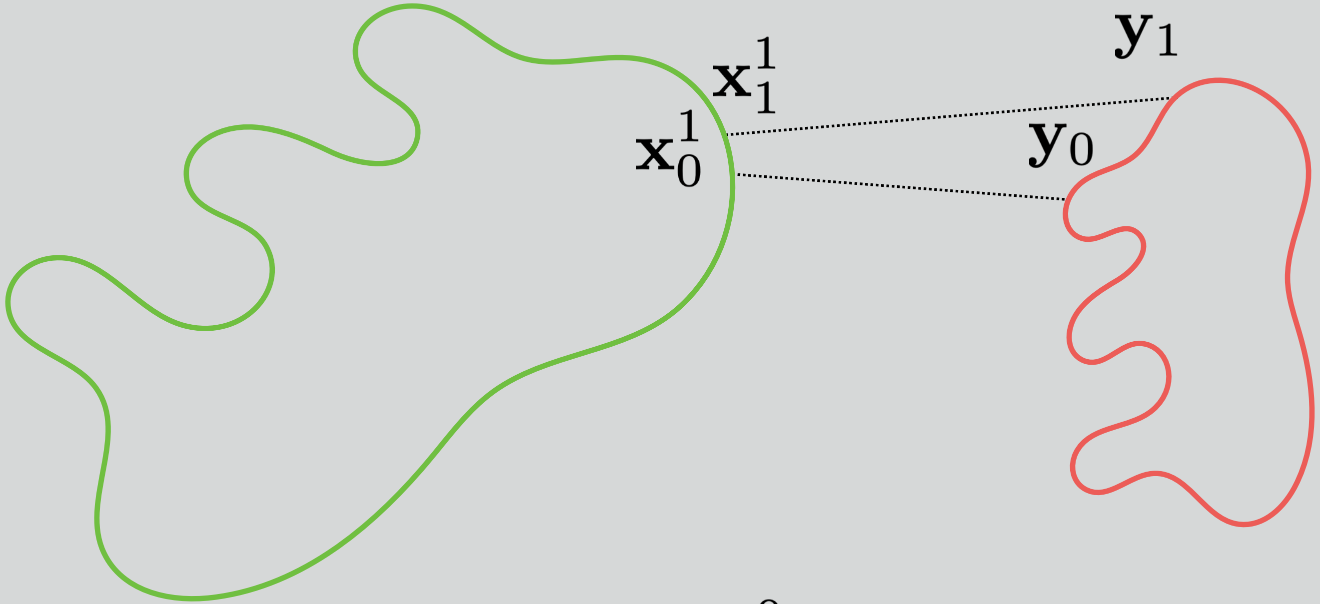


Neutral initialization. Initializing t to align centroids should work better!

$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \mathbf{0}, s = 1\}$$

$$\mathbf{x}_0^1 = \arg\min_{\mathbf{x} \in \mathbf{X}} \|f^0(\mathbf{x}) - \mathbf{y}_0\|^2$$

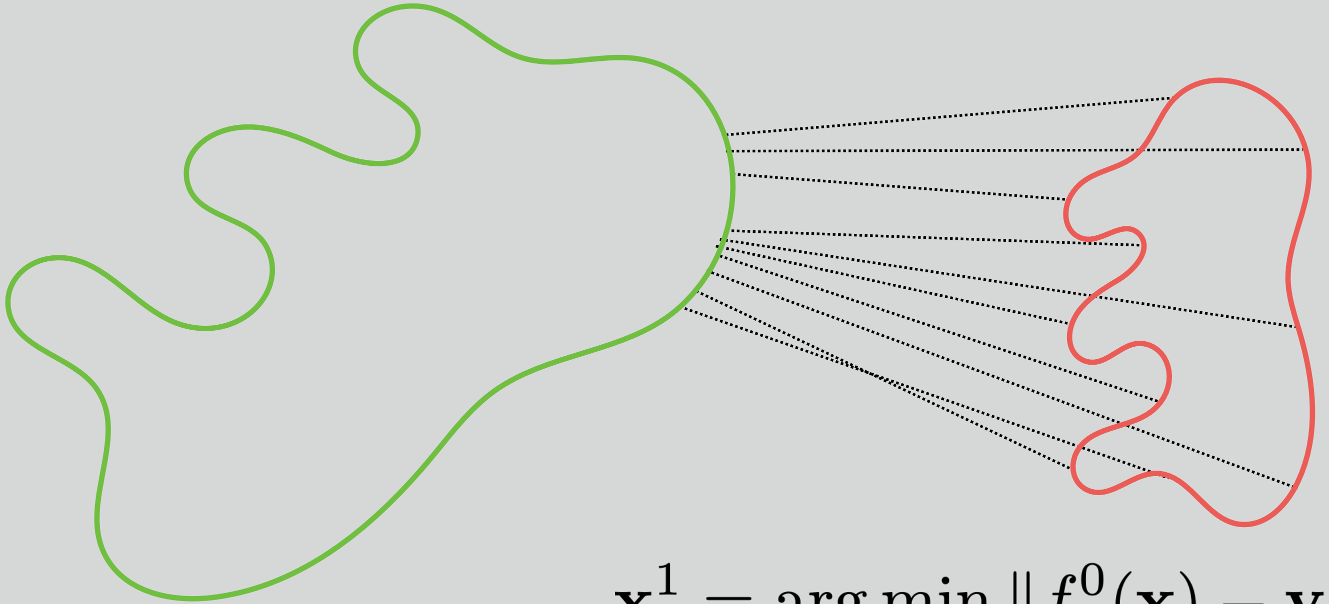
Make up reasonable correspondences



$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \mathbf{0}, s = 1\}$$

$$\mathbf{x}_i^1 = \arg\min_{\mathbf{x} \in \mathbf{X}} \|f^0(\mathbf{x}) - \mathbf{y}_i\|^2$$

Solve for the best transformation

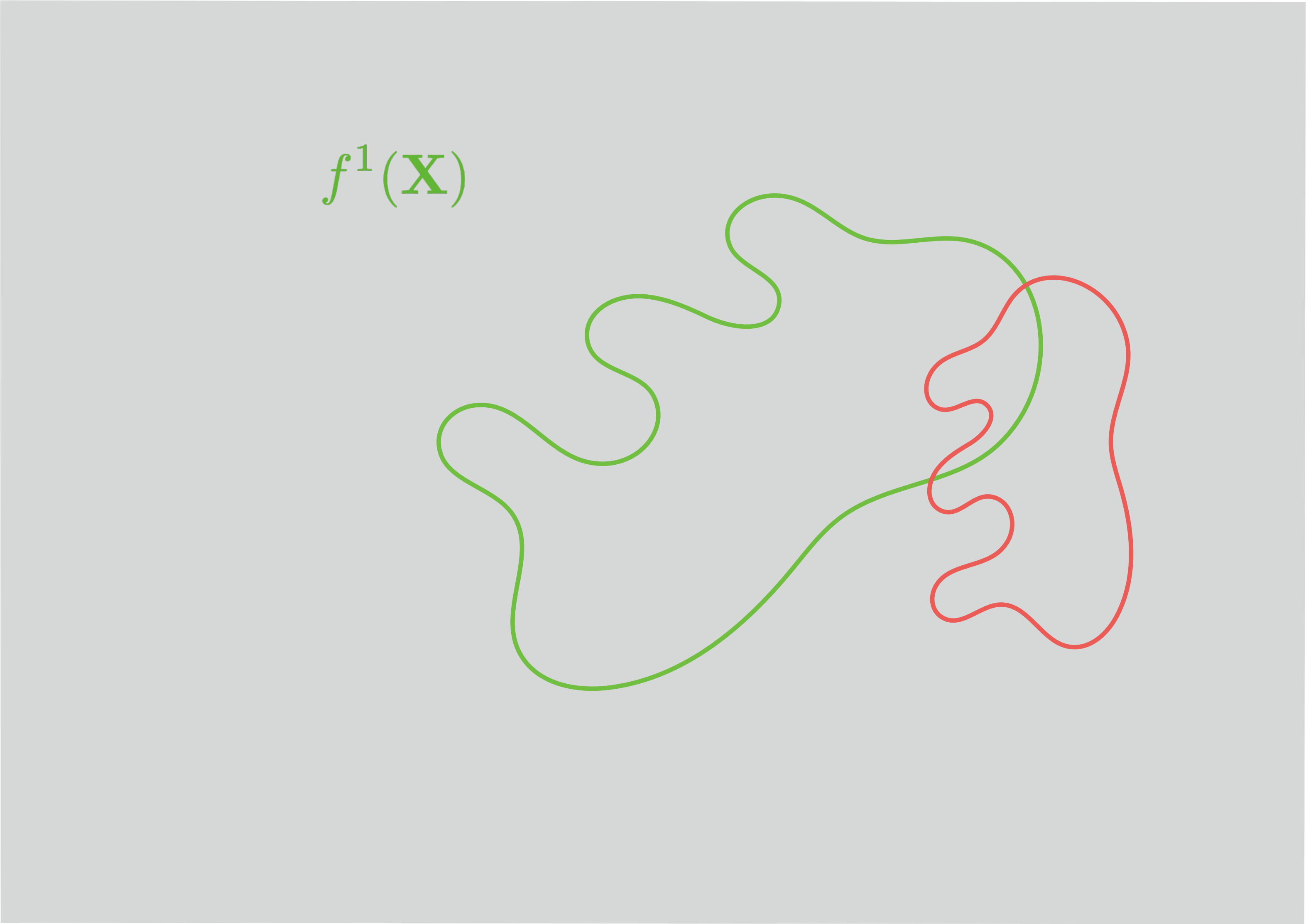


$$\mathbf{x}_i^1 = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^0(\mathbf{x}) - \mathbf{y}_i\|^2$$

solve with procrustes

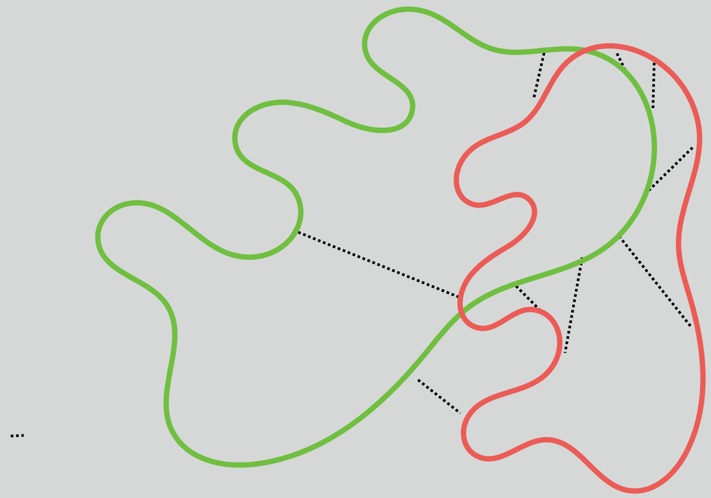
$$\boxed{f^1 = \arg\min_{f} \sum_i \|f(\mathbf{x}_i^1) - \mathbf{y}_i\|^2}$$

Apply it ...



and iterate!

$f^1(\mathbf{X})$

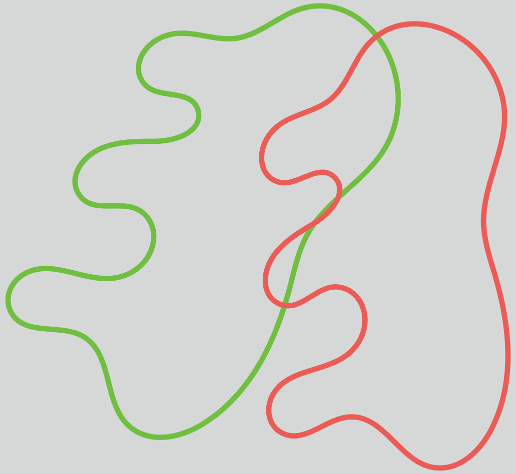


$$f^1 = \arg\min_f \sum_i \|f(\mathbf{x}_i^1) - \mathbf{y}_i\|^2$$

$$\mathbf{x}_i^2 = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^1(\mathbf{x}) - \mathbf{y}_i\|^2$$

and iterate!

$f^j(\mathbf{X})$



$$f^j = \arg\min_f \sum_i \|f(\mathbf{x}_i^j) - \mathbf{y}_i\|^2$$

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

and iterate!

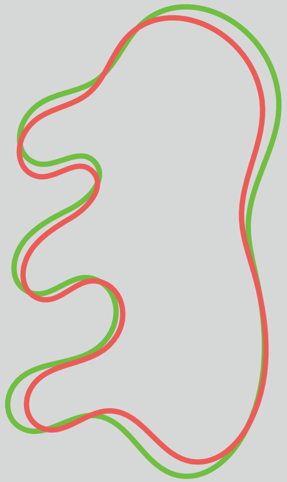
$f^j(\mathbf{X})$

$$f^j = \arg\min_f \sum_i \|f(\mathbf{x}_i^j) - \mathbf{y}_i\|^2$$

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

and iterate!

$f^j(\mathbf{X})$



$$f^j = \arg\min_f \sum_i \|f(\mathbf{x}_i^j) - \mathbf{y}_i\|^2$$

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

and iterate!

$$f^j = \arg\min_f \sum_i \| f(\mathbf{x}_i^j) - \mathbf{y}_i \|^2$$

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \| f^j(\mathbf{x}) - \mathbf{y}_i \|^2$$

Iterative Closest Point (ICP)

typically better than 0

1. Initialize

$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \boxed{\frac{\sum \mathbf{y}_i}{N} - \frac{\sum \mathbf{x}_i}{N}}, s = 1\}$$

Iterative Closest Point (ICP)

1. Initialize $$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \frac{\sum \mathbf{y}_i}{N} - \frac{\sum \mathbf{x}_i}{N}, s = 1\}$$

2. Compute correspondences according to current best transform
$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

Iterative Closest Point (ICP)

1. Initialize

$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \frac{\sum \mathbf{y}_i}{N} - \frac{\sum \mathbf{x}_i}{N}, s = 1\}$$

2. Compute correspondences according to current best transform

$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

3. Compute optimal transformation ($\mathbf{s}, \mathbf{R}, \mathbf{t}$)with Procrustes

$$f^{j+1} = \arg\min_f \sum_i \|f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i\|^2$$

Iterative Closest Point (ICP)

1. Initialize
$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \frac{\sum \mathbf{y}_i}{N} - \frac{\sum \mathbf{x}_i}{N}, s = 1\}$$

2. Compute correspondences according to current best transform
$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x}\in\mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$

3. Compute optimal transformation ($\mathrm{s}, \mathbf{R}, \mathbf{t}$)with Procrustes
$$f^{j+1} = \arg\min_f \sum_i \|f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i\|^2$$

4. Terminate if converged (error below a threshold), otherwise iterate

Iterative Closest Point (ICP)

1. Initialize
$$f^0 = \{\mathbf{R} = \mathbf{I}, \mathbf{t} = \frac{\sum \mathbf{y}_i}{N} - \frac{\sum \mathbf{x}_i}{N}, s = 1\}$$
2. Compute correspondences according to current best transform
$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \|f^j(\mathbf{x}) - \mathbf{y}_i\|^2$$
3. Compute optimal transformation ($\mathrm{s}, \mathbf{R}, \mathbf{t}$)with Procrustes
$$f^{j+1} = \arg\min_f \sum_i \|f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i\|^2$$
4. Terminate if converged (error below a threshold), otherwise iterate
5. Converges to local minima

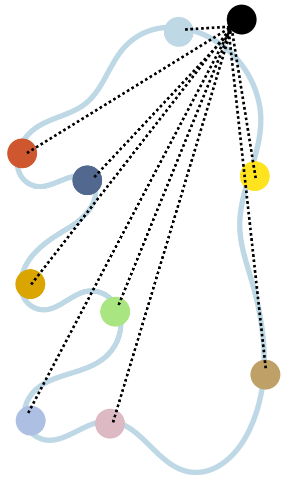
Is ICP the best we can do?

Iteration j:

- compute closest points
- compute optimal transformation with Procrustes
- apply transformation
- terminate if converged, otherwise iterate

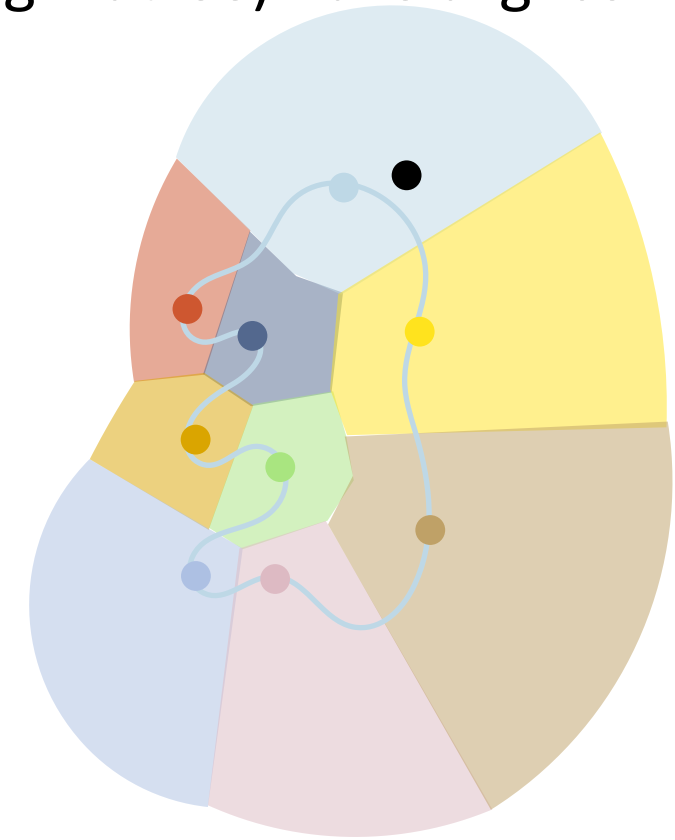
Closest points

- Brute force is $O(n^2)$
- For every source point find a neighbor point on the source shape



Closest points

- Tree based methods (e.g. kdtree) have avg. complexity log(n)

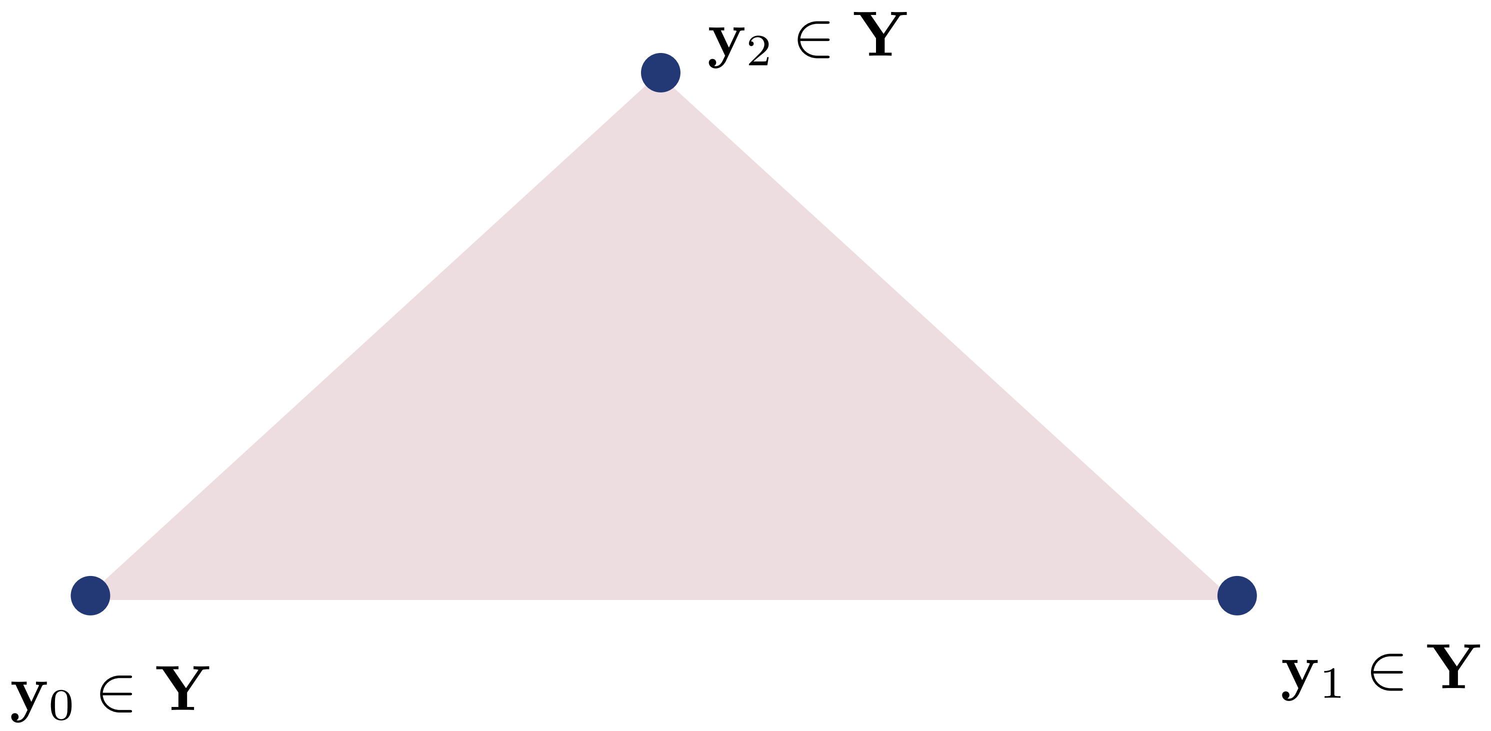


- Random point sampling also reduces the running time

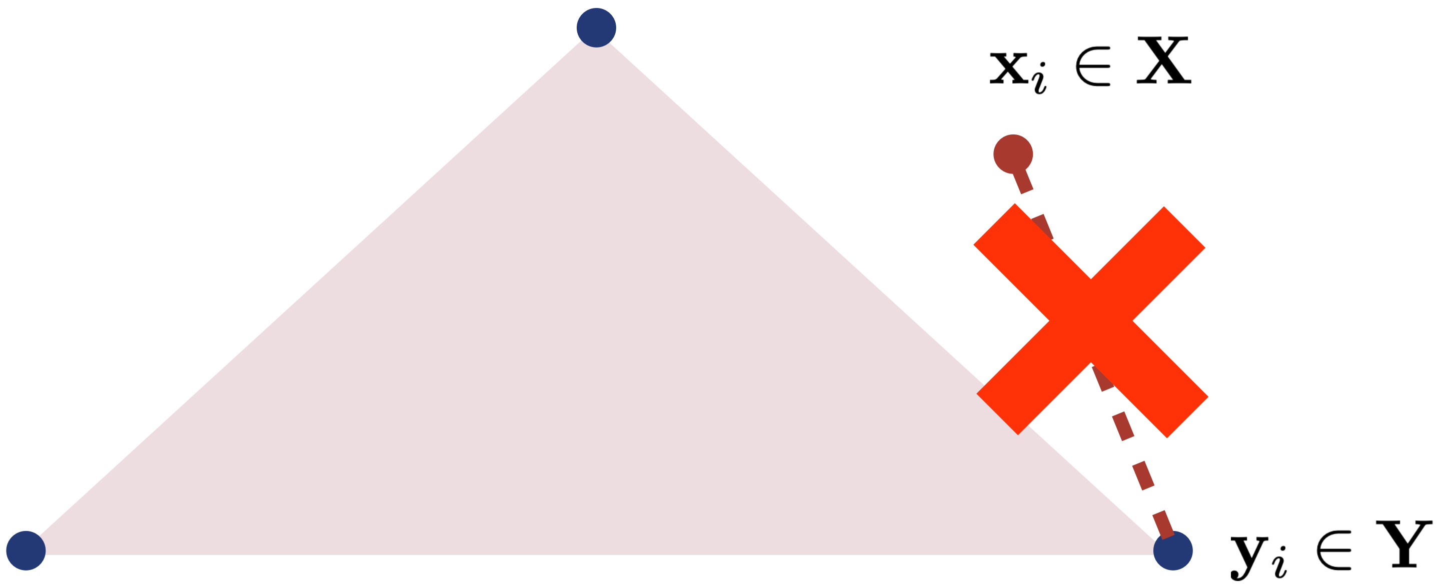
ICP: Tips to avoid local minima

- Always find correspondences from target to source!
  Proper **data term**
- Outliers —> Robust cost functions
- Use additional information (e.g. normals)
- Compute transformation based on greedy subsets of points: RANSAC

A much better objective: Point-to-surface distance

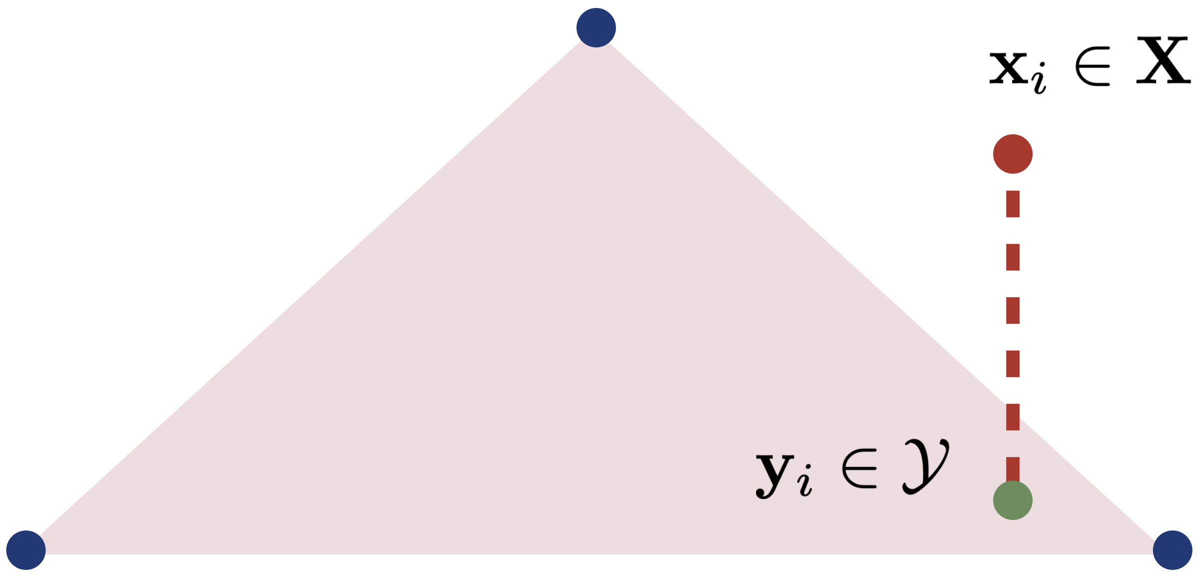


Closest points: avoid local minima



Point-to-point distance

Closest points: avoid local minima



Point-to-surface distance

Is ICP the best we can do?

Iteration j:

- compute closest points
- compute optimal transformation with Procrustes
- apply transformation
- terminate if converged, otherwise iterate

Best transformation?

- Procrustes gives us the optimal **rigid** transformation and scale given correspondences

- What if the deformation model is **not rigid** ?

- Can we generalise ICP to non-rigid deformation ?

Iterative Closest Point (ICP)

Iteration j:

- compute closest points ➔ Which direction to move?
- compute optimal transformation with Procrustes
- apply transformation
- terminate if converged, otherwise iterate

Iterative Closest Point (ICP)

Iteration j:

- compute closest points → Which direction to move?
- compute optimal transformation with Procrustes →
  Compute a transform that reduces the error
- apply transformation
- terminate if converged, otherwise iterate

Gradient-based ICP

Iteration j:

- compute closest points → Which direction to move?
- ~~compute optimal transformation with Procrustes~~ →

Compute descent step by linearising the energy

Jacobian of distance-based energy

- apply transformation
- terminate if converged, otherwise iterate

Gradient-based ICP

$$\boxed{\arg\min_f E(f) = \arg\min_f \sum_i \| f(\mathbf{x}_i^{j+1}) - \mathbf{y}_i \|^2}$$

- If f is a rigid transformation we can solve this minimisation using Procrustes

- If f is a general non-linear function ?
- Gradient descent: $f^{k+1} = f^k - \lambda \nabla_f E(f)$

- For least squares, is there a better optimisation method ?
  yes: Gauss-Newton based methods.

Gradient-based ICP

1. Energy:
$$E \equiv \sum_i \| \min_{\mathbf{x}} f(\mathbf{x}) - \mathbf{y}_i \|^2$$

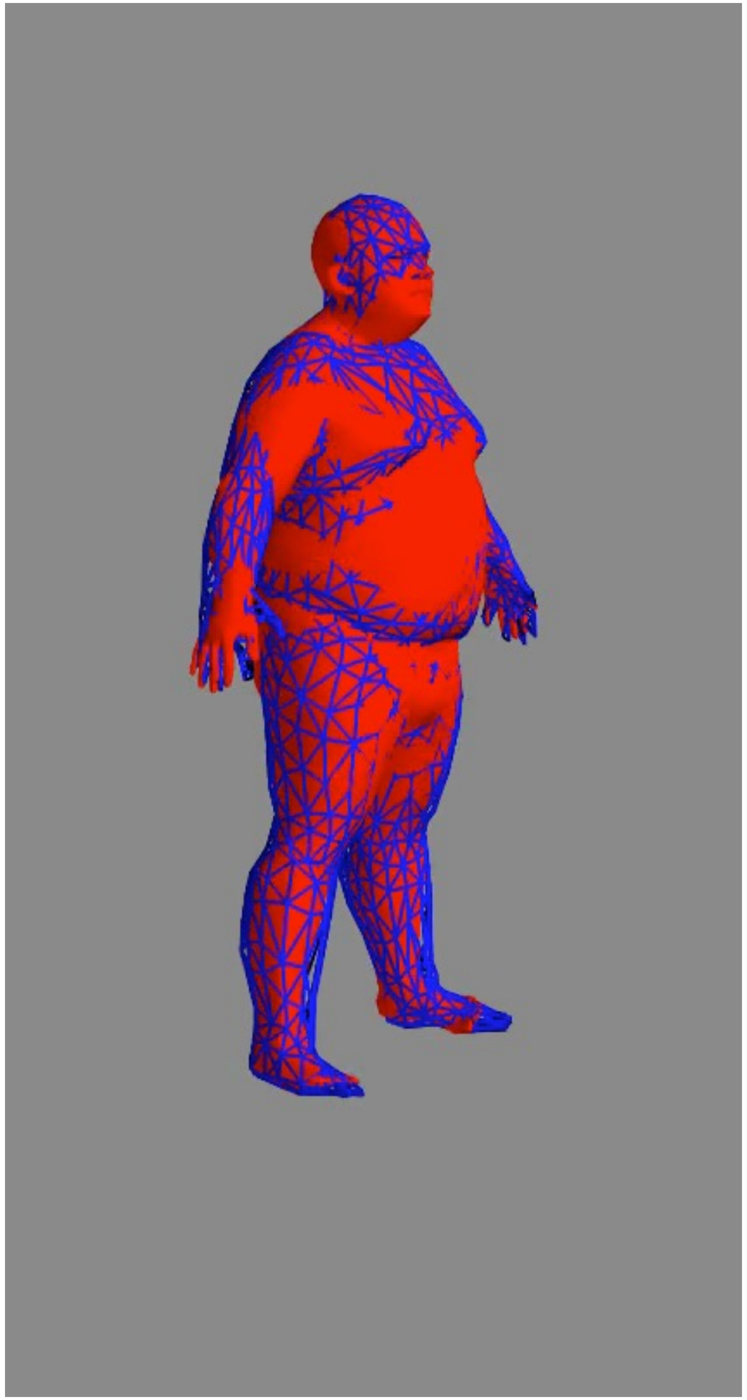
2. Consider the correspondences fixed in each iteration j+1
$$\mathbf{x}_i^{j+1} = \arg\min_{\mathbf{x} \in \mathbf{X}} \| f^j(\mathbf{x}) - \mathbf{y}_i \|^2$$

3. Compute gradient of the energy around current estimation
$$g^{j+1} = \nabla E(f^j)$$

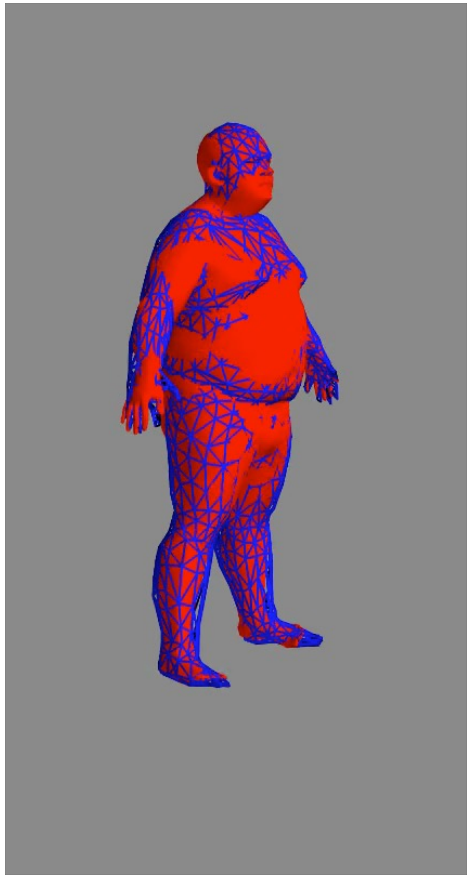
4. Apply step (gradient descent, dogleg, LM, BFGS…)
$$f^{j+1} = k_{step}(g^{0 \dots j+1}, f^{0 \dots j})$$
(for example $f^{j+1} = f^j - \alpha g^{j+1}$)

5. terminate if converged, otherwise iterate (go to step 2)

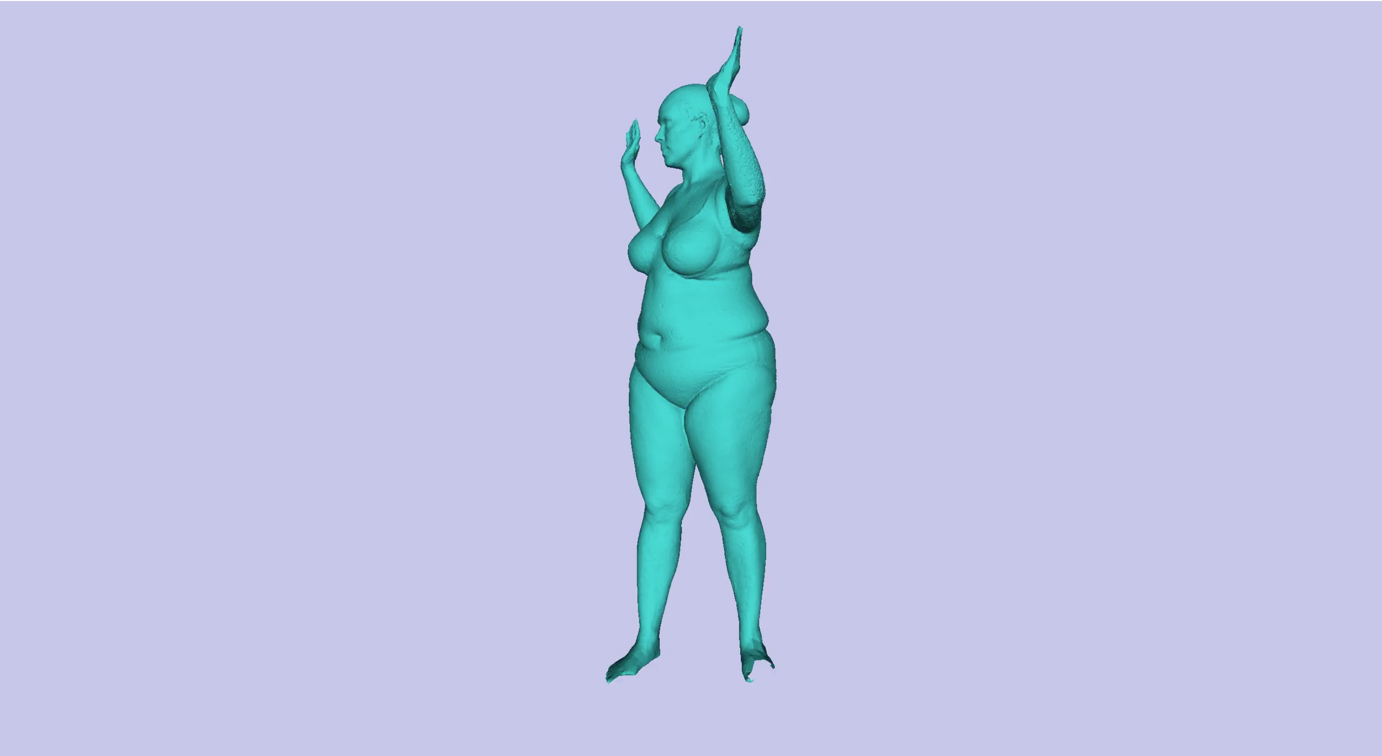
Gradient-based ICP



Why is convergence on the left less smooth?



Point to point objective



Point to surface objective

Gradient-based ICP

- Energy:
- Consider the correspondences fixed in each iteration j+1
- Compute gradient of the energy around current estimation
- Apply step (gradient descent, dogleg, LM, BFGS…)
- terminate if converged, otherwise iterate $f^{j+1} = k_{step}(g^{0\dots j+1}, f^{0\dots j})$

Gradient-based ICP

- Gradient: derivative of the sum of squared distances with respect to transformation f parameters

$$E \equiv \sum_i \| \min_{\mathbf{x}} f(\mathbf{x}) - \mathbf{y}_i \|^2$$

$$g^{j+1} = \nabla E(f^j)$$

Gradient-based ICP

- Gradient: derivative of the sum of squared distances with respect to transformation f parameters

$$E \equiv \sum_i \| \min_{\mathbf{x}} f(\mathbf{x}) - \mathbf{y}_i \|^2$$

$$g^{j+1} = \nabla E(f^j)$$

- Each derivative is easy
  - Who wants to writes it down?

Gradient-based ICP

- Gradient: derivative of the sum of squared distances with respect to transformation f parameters

$$E \equiv \sum_i \| \min_{\mathbf{x}} f(\mathbf{x}) - \mathbf{y}_i \|^2$$

$$g^{j+1} = \nabla E(f^j)$$

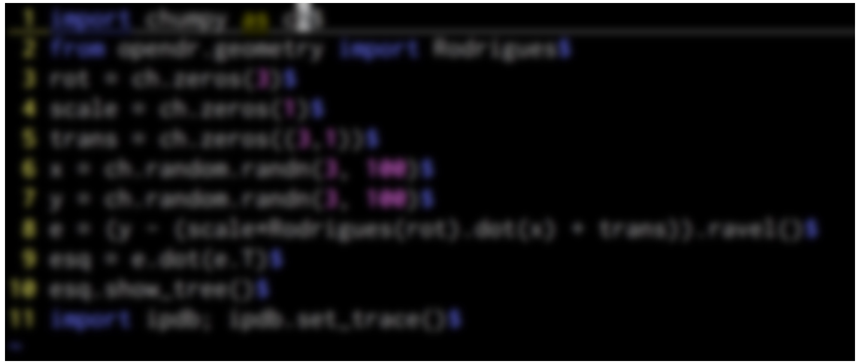
- Each derivative is easy
  - Who wants to writes it down?

- **Chain rule and automatic differentiation!**

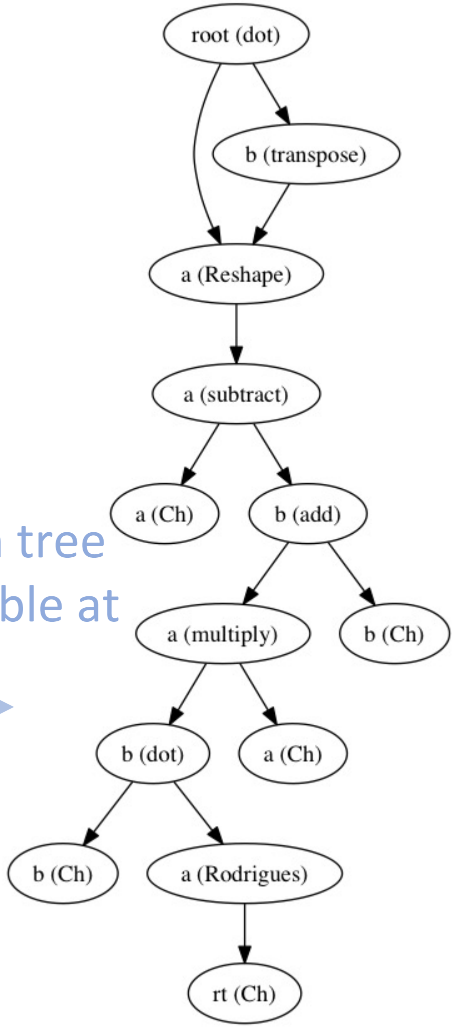
Automatic differentiation

$$E = \sum_i \|s\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2$$

write as if it was numpy code



results in expression tree with jacobians available at each step



Gradient-based ICP

- Energy:
- Consider the correspondences fixed in each iteration j+1
- Compute gradient of the energy around current estimation
- Apply step (gradient descent, dogleg, LM, BFGS…)
- terminate if converged, otherwise iterate $f^{j+1} = k_{step}(g^{0\dots j+1}, f^{0\dots j})$

Why Gradient-based ICP?

- Formulation is much more generic: the energy can incorporate other terms, more parameters, etc
- A lot of available software for solving this least squares problem (cvx, ceres, …)
- **However,** the resulting energy is non-convex for general deformation models. Optimisation can get trapped in local minima.

Take-home message

- **Procrustes** is **optimal** for **rigid alignment problems** with *known* correspondences. For other problems:

- We can compute **correspondences** and solve for the best **transformation** iteratively with Iterative Closest Point (**ICP**)

Slide credits

- Javier Romero