Probabilistic Robotics Course

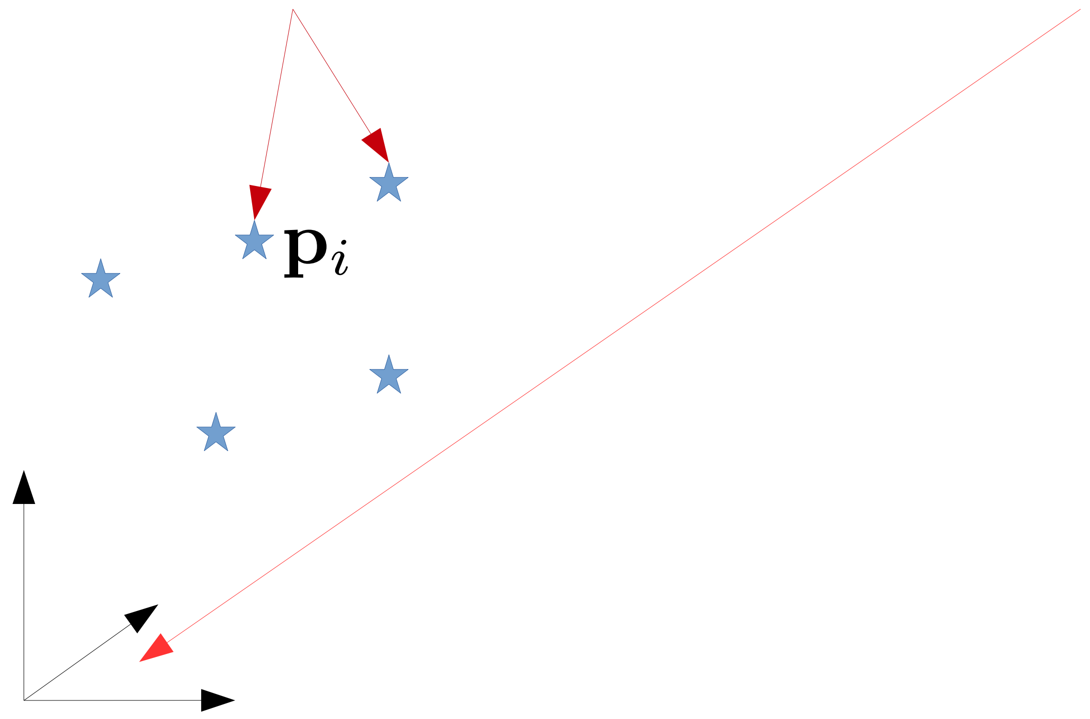
3D Point Registration

G. Grisetti
B. Della Corte
grisetti@dis.uniroma1.it

Dept of Computer Control and Management Engineering
Sapienza University of Rome

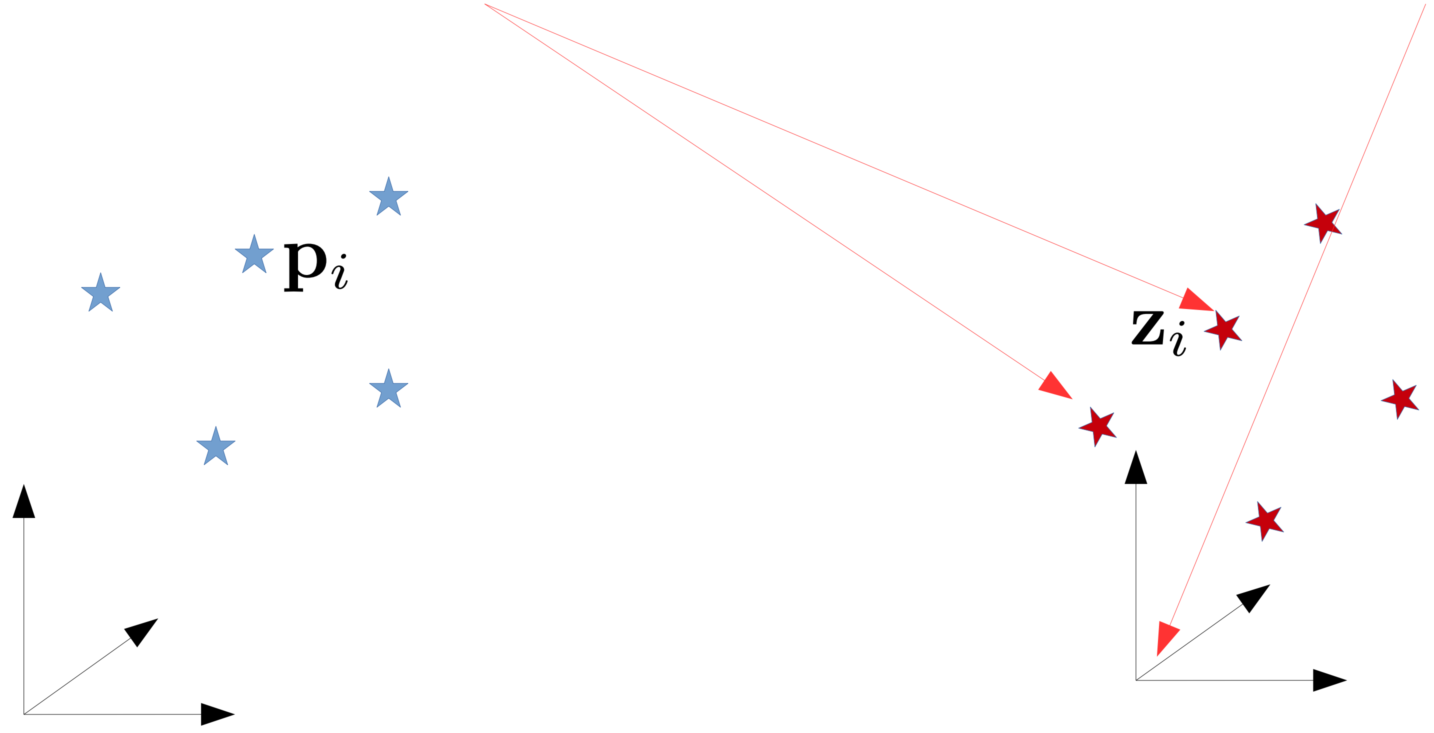
Example ICP Optimization in 3D

Given a set of points in the world frame



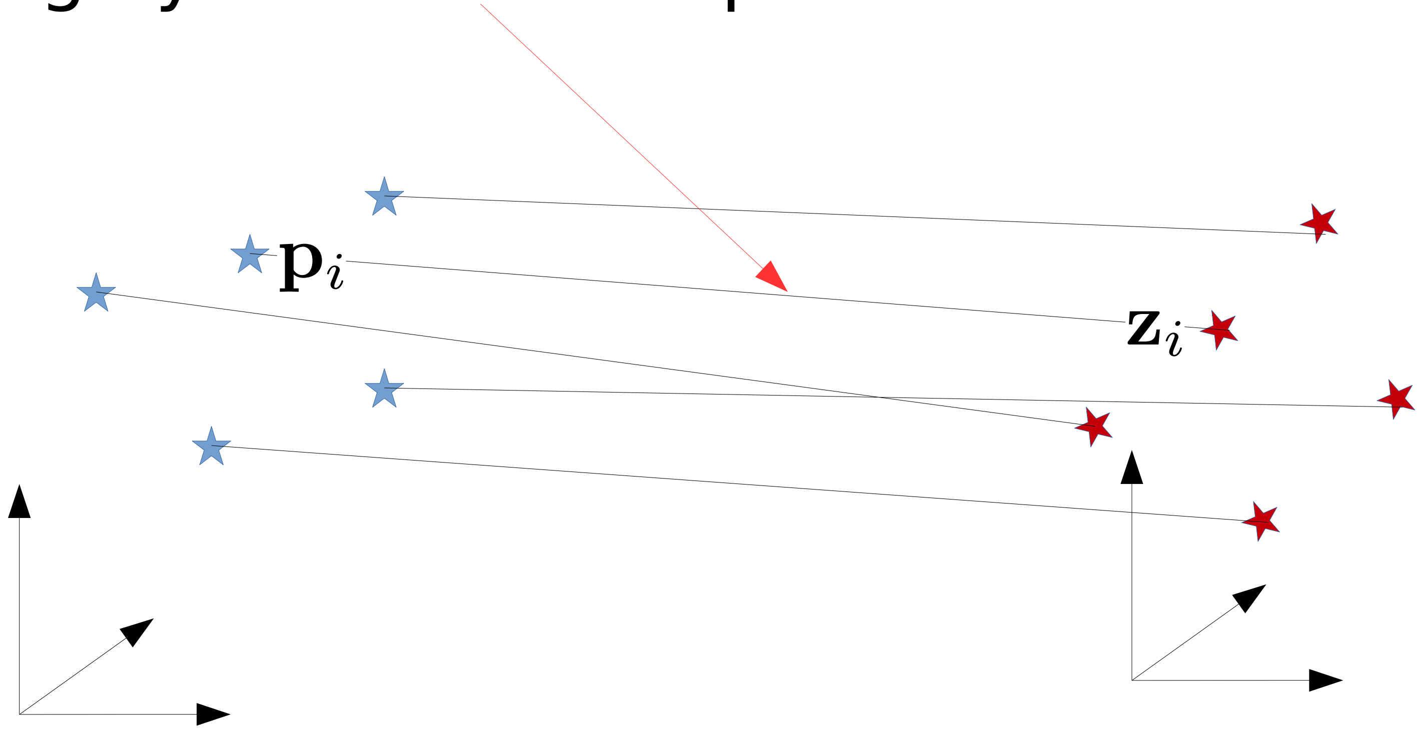
Example ICP Optimization in 3D

A set of 3D measurements in the robot frame



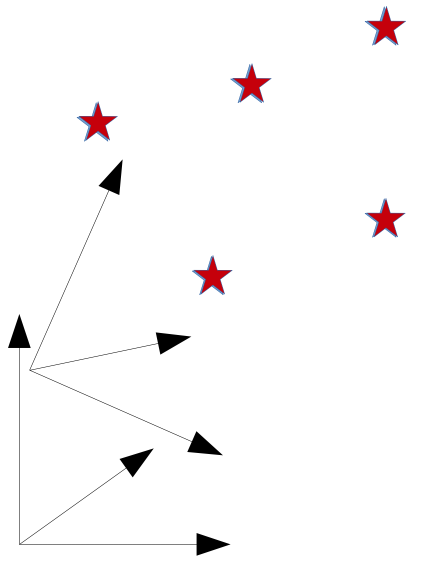
Example ICP Optimization in 3D

Roughly known correspondences



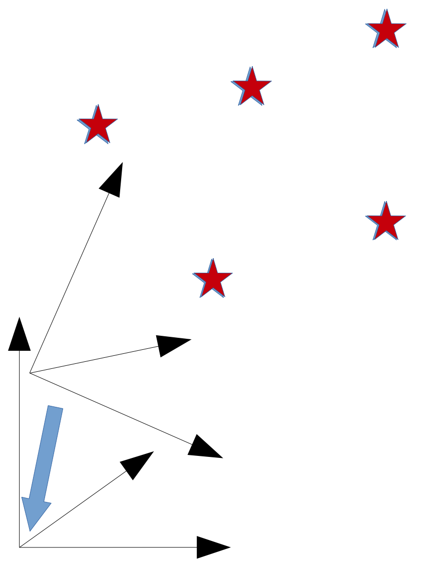
Example ICP Optimization in 3D

We want to find a transform that minimizes distance between corresponding points



Example ICP Optimization in 3D

Such a transform will be the pose of world w.r.t. robot



Note: we can also estimate robot w.r.t world, but it leads to longer calculations

ICP: State and Measurements

State

$$\begin{aligned} \mathbf{x} &\in SE(3) \\ \mathbf{x} &= (\underbrace{x\ y\ z}_{\mathbf{t}}\ \underbrace{\alpha_x\ \alpha_y\ \alpha_z}_{\alpha})^T \end{aligned}$$

Measurements

$$\begin{aligned} \mathbf{z} &\in \Re^3 \\ \mathbf{h}^{[i]}(\mathbf{x}) &= \mathbf{R}(\alpha)\mathbf{p}^{[i]} + \mathbf{t} \end{aligned}$$

On Rotation Matrices

A rotation is obtained by composing the rotations along *x-y-z*

$$\mathbf{R}(\alpha) \quad = \quad \mathbf{R}_x(\alpha_x)\mathbf{R}_y(\alpha_y)\mathbf{R}_z(\alpha_z)$$

Small lookup of rotations (and derivatives)

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c & -s \\ 0 & s & c \end{pmatrix} \quad \mathbf{R}_y = \begin{pmatrix} c & 0 & s \\ 0 & 1 & 0 \\ -s & 0 & c \end{pmatrix} \quad \mathbf{R}_z = \begin{pmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{R}'_x = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -s & -c \\ 0 & c & -s \end{pmatrix} \quad \mathbf{R}'_y = \begin{pmatrix} -s & 0 & c \\ 0 & 0 & 0 \\ -c & 0 & -s \end{pmatrix} \quad \mathbf{R}'_z = \begin{pmatrix} -s & -c & 0 \\ c & -s & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

ICP: Jacobian

$$\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \mathbf{t}} \; \frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_x} \; \frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_y} \; \frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_z} \right)$$

$$\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \mathbf{t}} = \mathbf{I}$$

$$\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_x} = \mathbf{R}'_x \mathbf{R}_y \mathbf{R}_z \mathbf{p}^{[i]}$$

$$\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_y} = \mathbf{R}_x \mathbf{R}'_y \mathbf{R}_z \mathbf{p}^{[i]}$$

$$\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \alpha_z} = \mathbf{R}_x \mathbf{R}_y \mathbf{R}'_z \mathbf{p}^{[i]}$$

ICP: Octave Code

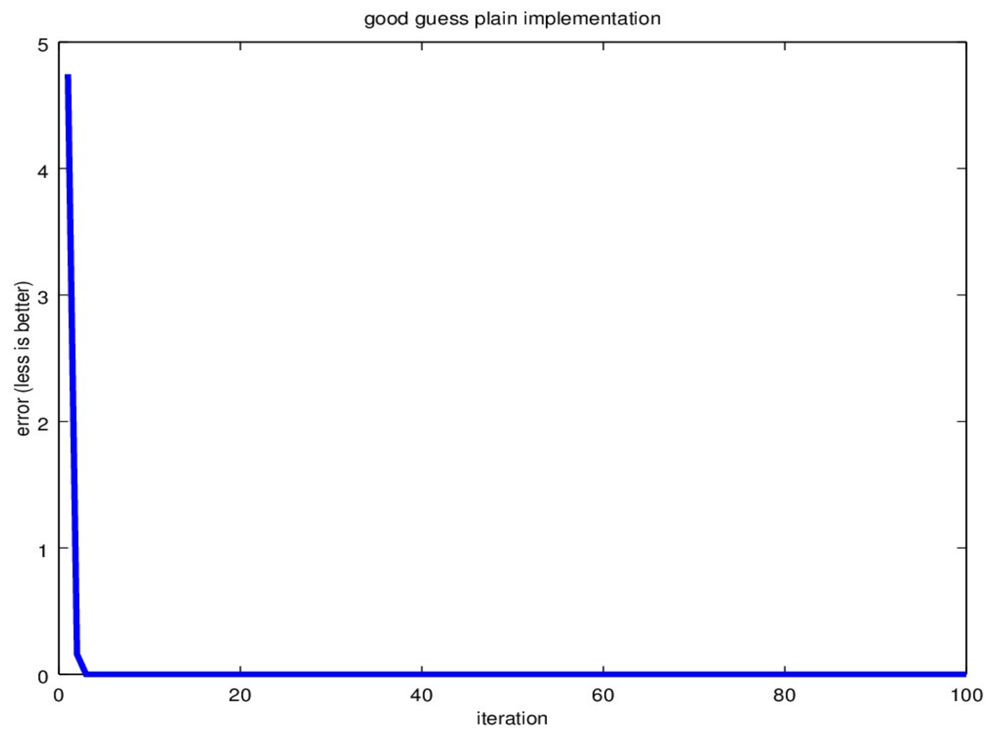
```
function [e,J]=errorAndJacobian(x,p,z)
  rx=Rx(x(4)); #rotation matrices at x
  ry=Ry(x(5));
  rz=Rz(x(6));
  rx_p=Rx_prime(x(4)); #derivatives at x
  ry_p=Ry_prime(x(5));
  rz_p=Rz_prime(x(6));
  t=x(1:3);

  z_hat=rx*ry*rz*p+t; #prediction
  e=z_hat-z;          #error
  J=zeros(3,6);       #jacobian
  J(1:3,1:3)=eye(3);  #translational part of jacobian


  J(1:3,4)=rx_p*ry*rz*p; #de/dax
  J(1:3,5)=rx*ry_p*rz*p; #de/day
  J(1:3,6)=rx*ry*rz_p*p; #de/daz
endfunction
```

ICP: Octave Code

```
function [x, chi_stats]=doICP(x_guess, P, Z,
num_iterations)
  x=x_guess;
  chi_stats=zeros(1,num_iterations); #ignore this for
now
  for (iteration=1:num_iterations)
    H=zeros(6,6);
    b=zeros(6,1);
    chi=0;
    for (i=1:size(P,2))
      [e,J] = errorAndJacobian(x, P(:,i), Z(:,i));
      H+=J'*J;
      b+=J'*e;
      chi+=e'*e;
    endfor
    chi_stats(iteration)=chi;
    dx=-H\b;
    x+=dx;
  endfor
endfunction
```

Testing, good initial guess

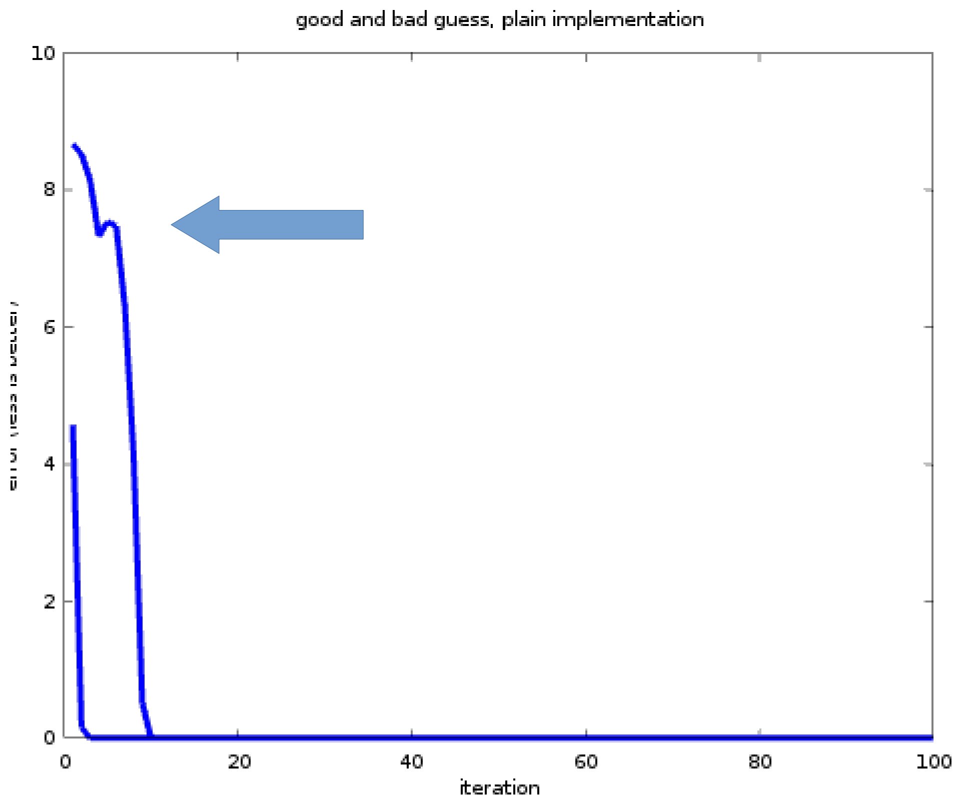
- Spawn a set of random points in 3D
- Define a location of the robot
- Compute synthetic measurements from that location
- Set the a point close to the true location as initial guess
- Run ICP and plot the evolution of the error



- When started from a good guess, the system converges nicely

Testing, bad initial guess

- Spawn a set of random points in 3D
- Define a location of the robot
- Compute synthetic measurements from that location
- Set the origin as initial guess
- Run ICP and plot the evolution of the error



- If the guess is poor, the system might take long to converge
- The error might increase

ICP Linear Relaxation

Can we approach the problem without iteration?

- The rotations in the prediction function make the Jacobian dependent on the initial guess
- We can relax the constraint that R is a rotation matrix and estimate 12 parameters of an affine transformation that minimizes the error

$$\mathbf{x}^T = (\mathbf{r}_1^T \ \mathbf{r}_2^T \ \mathbf{r}_3^T \ \mathbf{t}^T)$$

$$\begin{aligned}
\mathbf{h}^{[i]}(\mathbf{x}) &= \mathbf{R}\mathbf{p}^{[i]} + \mathbf{t} \\
&= \begin{pmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{pmatrix} \mathbf{p}^{[i]} + \mathbf{t} \\
&= \underbrace{\begin{pmatrix} \mathbf{p}^{[i]T} & & \\ & \mathbf{p}^{[i]T} & \\ & & \mathbf{p}^{[i]T} \end{pmatrix}}_{\mathbf{M}^{[i]}} \begin{pmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{pmatrix} + \mathbf{t}
\end{aligned}$$

ICP Linear Relaxation

Jacobians

$$
\begin{aligned}
\mathbf{x}^T &= (\mathbf{r}_1^T \ \mathbf{r}_2^T \ \mathbf{r}_3^T \ \mathbf{t}^T) \\
\mathbf{h}^{[i]}(\mathbf{x}) &= \mathbf{M}^{[i]} \begin{pmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \mathbf{r}_3 \end{pmatrix} + \mathbf{t} \\
\frac{\partial \mathbf{h}^{[i]}(\mathbf{x})}{\partial \mathbf{x}} &= \left(\mathbf{M}^{[i]} \mid \mathbf{I} \right)
\end{aligned}
$$

3x12 matrix not depending on **x.** Minimum found in 1 iteration of least squares

If the initial transform is the Identity, and Omega=I, the least squares problem reduces to the following system

$$
\begin{aligned}
\mathbf{H} &= \begin{pmatrix} \sum_i \mathbf{M}^{[i]T}\mathbf{M}^{[i]} & \sum_i \mathbf{M}^{[i]^T} \\ \sum_i \mathbf{M}^{[i]} & \sum_i \mathbf{I} \end{pmatrix} \\
\mathbf{b} &= \sum_i \left(\mathbf{M}^{[i]} \mid \mathbf{I} \right)^T (\mathbf{h}^{[i]}(\mathbf{x}) - \mathbf{z}^{[i]})
\end{aligned}
$$

ICP Linear Relaxation

Applying one iteration of least squares, we can find 12 parameters that minimize the error.

- The vectors $\mathbf{r}_1$,$\mathbf{r}_2$ and $\mathbf{r}_3$ do are not orthonormal!
- We need to find the rotation matrix "closer" to the solution reported by least squares
- This can be done by SVD decomposition

Linear part of the affine transform returned by least squares

Diagonal matrix. If Identity, A is a rotation matrix

$$\mathbf{A} = \begin{pmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{pmatrix} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

Orthonormal matrices

Rotation matrix "close" to **A**
(if D close to I)

$$\mathbf{R} = \mathbf{U}\mathbf{V}^T$$

If D far from "I", the solution reported through linear relaxation is not good.

ICP Linear Relaxation

Linear relaxation provides a reasonable solution when

- There are few wrong correspondences
- The error is small

Violation to these assumption result in an affine transformation that is not an Isometry, and uses the additional degrees of freedom to reduce the error.

You can check this condition by inspecting the values of the D matrix.

Few further iterations of non-linear least squares are usually beneficial to refine the solution.