Probabilistic Robotics Course

ICP optimization on a Manifold

Giorgio Grisetti

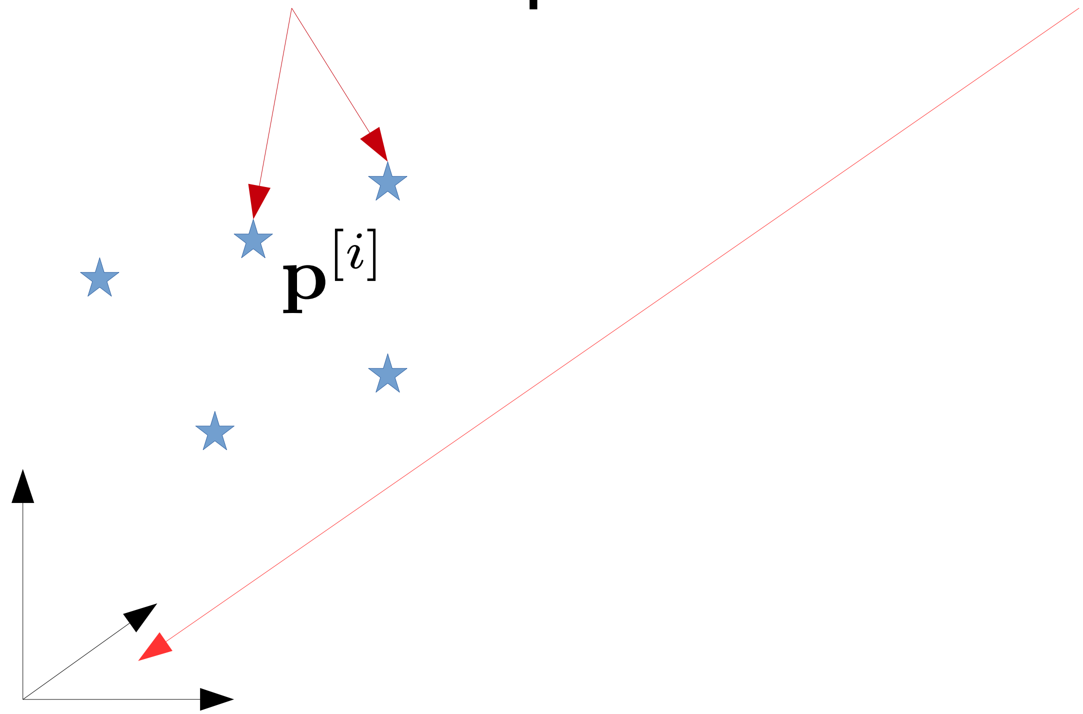
grisetti@diag.uniroma1.it

Department of Computer, Control, and Management Engineering
Sapienza University of Rome

Courtesy of Li et. al

Example ICP Optimization 3D

Given a set of points in the world frame

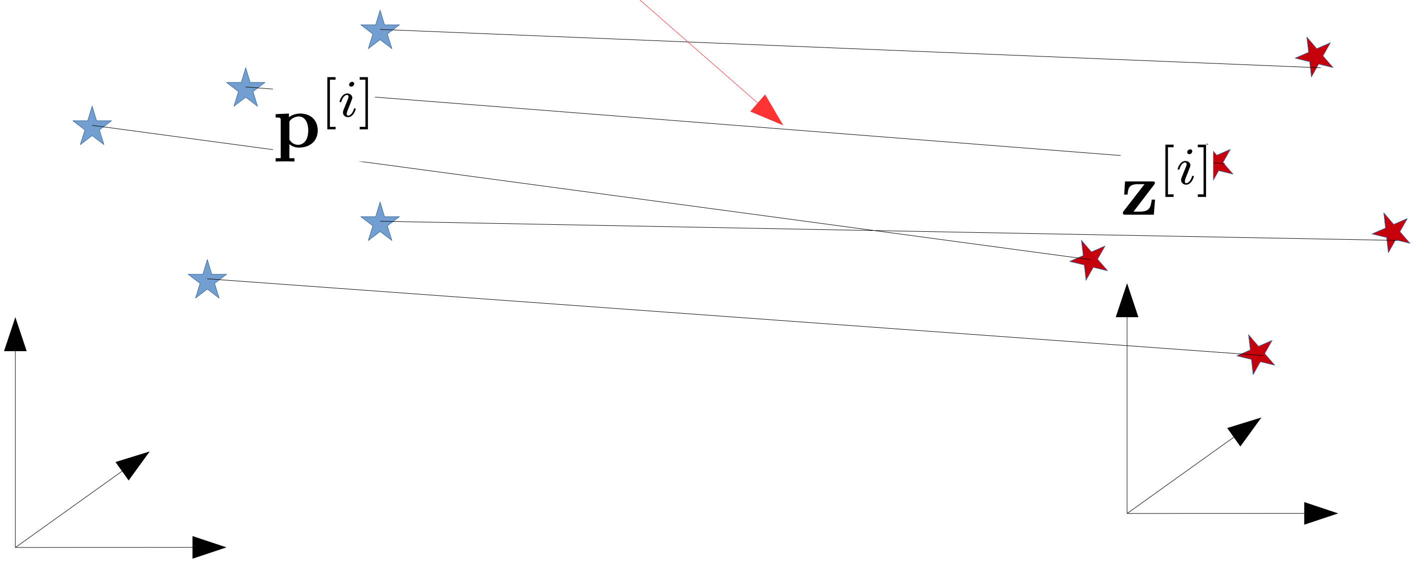


Example ICP Optimization 3D

A set of 3D measurements in the robot frame

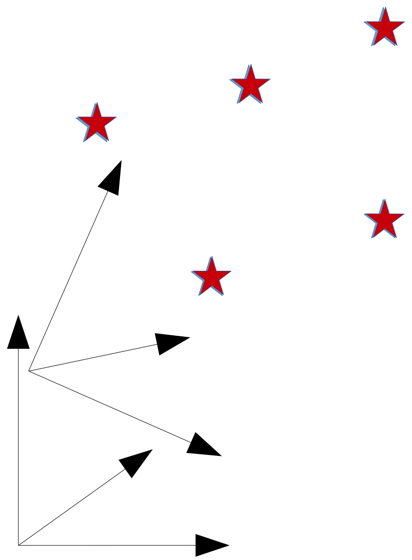
Example ICP Optimization 3D

Roughly known correspondences



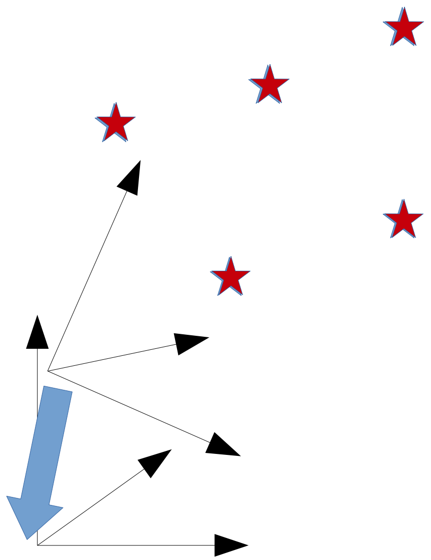
Example ICP Optimization 3D

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



Example ICP Optimization 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

Algorithm (One Iteration)

Clear **H** and **b**

$$\mathbf{H} \leftarrow 0 \qquad \mathbf{b} \leftarrow 0$$

For each measurement

$$\begin{aligned}
\mathbf{e}_i &\leftarrow \mathbf{h}^{[i]}(\mathbf{X}^*) \boxminus \mathbf{Z}^{[i]} \\
\mathbf{J}^{[i]} &\leftarrow \left.\frac{\partial \mathbf{e}^{[i]}(\mathbf{X}^* \boxplus \mathbf{\Delta x})}{\partial \mathbf{\Delta x}}\right|_{\mathbf{\Delta x}=\mathbf{0}} \\
\mathbf{H} &\leftarrow \mathbf{H} + \mathbf{J}^{[i]T} \mathbf{\Omega}^{[i]} \mathbf{J}^{[i]} \\
\mathbf{b} &\leftarrow \mathbf{b} + \mathbf{J}^{[i]T} \mathbf{\Omega}^{[i]} \mathbf{e}^{[i]}
\end{aligned}$$

Compute and apply the perturbation

$$\begin{aligned}
\mathbf{\Delta x} &\leftarrow \mathrm{solve}(\mathbf{H}\mathbf{\Delta x} = -\mathbf{b}) \\
\mathbf{X}^* &\leftarrow \mathbf{X}^* \boxplus \mathbf{\Delta x}
\end{aligned}$$

Methodology

State space **X**

- Qualify the Domain
- Define a Euclidean parameterization for the perturbation
- Define boxplus operator

Measurement space(s) **Z**

- Qualify the Domain
- Define a Euclidean parameterization for the perturbation
- Define boxminus operator

Identify the prediction functions **h(X)**

MICP: State and Measurements

State

$$\begin{aligned}
\mathbf{X} &= [\mathbf{R}|\mathbf{t}] \in SE(3) \\
\boldsymbol{\Delta}\mathbf{x} &= (\underbrace{\Delta x \; \Delta y \; \Delta z}_{\boldsymbol{\Delta}\mathbf{t}} \; \underbrace{\Delta \alpha_x \; \Delta \alpha_y \; \Delta \alpha_z}_{\boldsymbol{\Delta}\alpha})^T \\
\mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{x} &= \mathrm{v2t}(\boldsymbol{\Delta}\mathbf{x})\mathbf{X} \\
&= [\mathbf{R}(\boldsymbol{\Delta}\alpha)\mathbf{R}|\mathbf{R}(\boldsymbol{\Delta}\alpha)\mathbf{t} + \boldsymbol{\Delta}\mathbf{t}]
\end{aligned}$$

Measurements

$$\begin{aligned}
\mathbf{z} &\in \Re^3 \\
\mathbf{h}^{[i]}(\mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{x}) &= (\mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{x})\mathbf{p}^{[i]} = \mathbf{R}(\boldsymbol{\Delta}\alpha) \underbrace{\left[\mathbf{R}\mathbf{p}^{[i]} + \mathbf{t}\right]}_{\mathbf{p'}^{[i]}} + \boldsymbol{\Delta}\mathbf{t}
\end{aligned}$$

MICP: Error

The measurements are Euclidean, no need for boxminus

$$\begin{aligned}\mathbf{e}^{[i]}(\mathbf{X} \boxplus \mathbf{\Delta x}) &= \mathbf{h}^{[i]}(\mathbf{X} \boxplus \mathbf{\Delta x}) - \mathbf{z}^{[i]} \\ &= \mathbf{R}(\mathbf{\Delta}\alpha)\mathbf{p}'^{[i]} + \mathbf{\Delta t} - \mathbf{z}^{[i]}\end{aligned}$$

On Rotation Matrices

A rotation is obtained by composing the rotations along *x-y-z* $\mathbf{R}(\alpha) \quad = \quad \mathbf{R}_x(\alpha_x)\mathbf{R}_y(\alpha_y)\mathbf{R}_z(\alpha_z)$

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

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

$$\mathbf{R}'_{x=0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \quad \mathbf{R}'_{y=0} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \quad \mathbf{R}'_{z=0} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

MICP: Jacobian

Linearizing around the **0** of the chart simplifies the calculations

$$
\begin{aligned}
\mathbf{J}^{[i]} &= \left.\frac{\partial \mathbf{e}^{[i]}(\mathbf{X} \boxplus \mathbf{\Delta x})}{\partial \mathbf{\Delta x}}\right|_{\mathbf{\Delta x}=0} \\
&= \left.\left(\frac{\partial \mathbf{e}^{[i]}(\cdot)}{\partial \mathbf{\Delta t}} \;\; \frac{\partial \mathbf{e}^{[i]}(\cdot)}{\partial \mathbf{\Delta} \alpha} \right)\right|_{\mathbf{\Delta x}=0} \\
&= \left.\left(\frac{\partial \mathbf{\Delta t}}{\partial \mathbf{\Delta t}} \;\; \frac{\partial \mathbf{R}(\mathbf{\Delta}\alpha)\mathbf{p}'^{[i]}}{\partial \mathbf{\Delta}\alpha} \right)\right|_{\mathbf{\Delta x}=0} \\
&= \left.\left(\frac{\partial \mathbf{\Delta t}}{\partial \mathbf{\Delta t}} \;\; \frac{\partial \mathbf{R}(\mathbf{\Delta}\alpha_{\mathbf{x}})\mathbf{p}'^{[i]}}{\partial \mathbf{\Delta}\alpha_{\mathbf{x}}} \; \frac{\partial \mathbf{R}(\mathbf{\Delta}\alpha_{\mathbf{y}})\mathbf{p}'^{[i]}}{\partial \mathbf{\Delta}\alpha_{\mathbf{y}}} \; \frac{\partial \mathbf{R}(\mathbf{\Delta}\alpha_{\mathbf{z}})\mathbf{p}'^{[i]}}{\partial \mathbf{\Delta}\alpha_{\mathbf{z}}} \right)\right|_{\mathbf{\Delta x}=0} \\
&= \left(\mathbf{I} \;\; -\left[\mathbf{p}'^{[i]}\right]_{\times} \right)
\end{aligned}
$$

MICP: Code

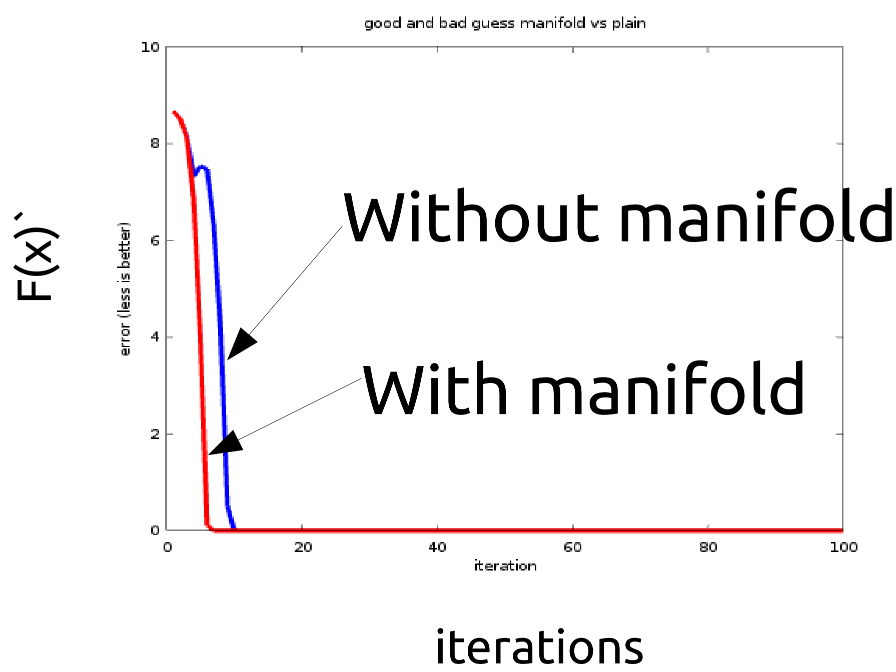
```
function T=v2t(v)
    T=eye(4);
    T(1:3,1:3)=Rx(v(4))*Ry(v(5))*Rz(v(6));
    T(1:3,4)=v(1:3);
endfunction;


function [e,J]=errorAndJacobianManifold(X,p,z)
   z_hat=X(1:3,1:3)*p+X(1:3,4); #prediction
   e=z_hat-z;
   J=zeros(3,6);
   J(1:3,1:3)=eye(3);
   J(1:3,4:6)=-skew(z_hat);
endfunction
```

MICP: Code

```
function [X, chi_stats]=doICPManifold(X_guess, P, Z, n_it)
  X=X_guess;
  chi_stats=zeros(1,n_it);
  for (iteration=1:n_it)
    H=zeros(6,6);
    b=zeros(6,1);
    chi=0;
    for (i=1:size(P,2))
      [e,J] = errorAndJacobianManifold(X, P(:,i), Z(:,i));
      H+=J'*J;
      b+=J'*e;
      chi+=e'*e;
    endfor
    chi_stats(iteration)=chi;
    dx=-H\b;
    X=v2t(dx)*X;
  endfor
endfunction
```

Testing

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



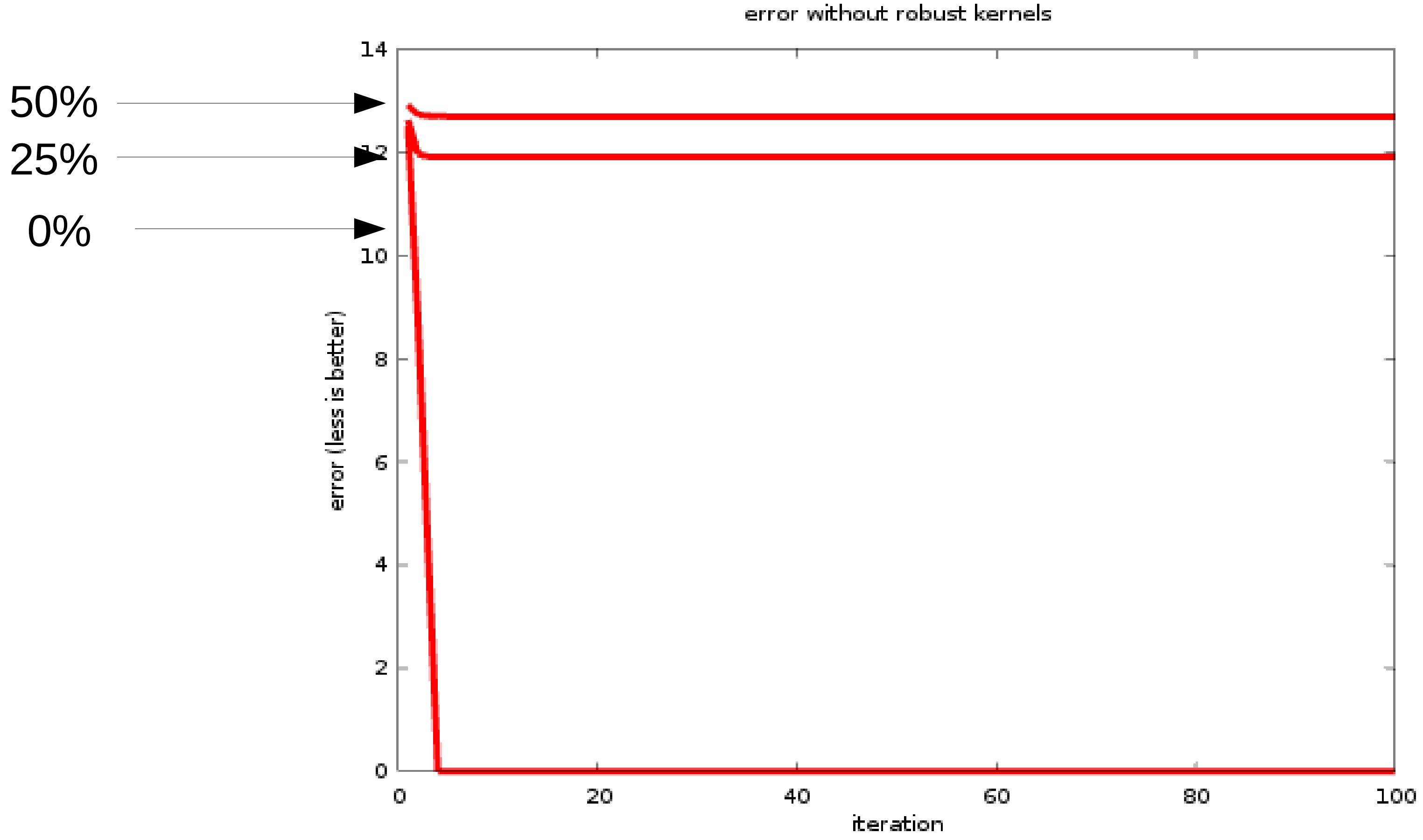
I need about 5 iterations to get a decent error

Outliers

Data association might fail generating false associations: the corresponding measurements are called **outliers.**

Good points are called **inliers**.

Let's see what happens when we inject an increasing number of outliers.



Robust Kernels

There will be outliers

Hint: Lessen the contribution of measurements having higher error (e.g. using Robust Kernels)

Trivial Kernel Implementation:

```
If (error>threshold){
    scale_error_so_that_its_norm_is_the_threshold();
}
```

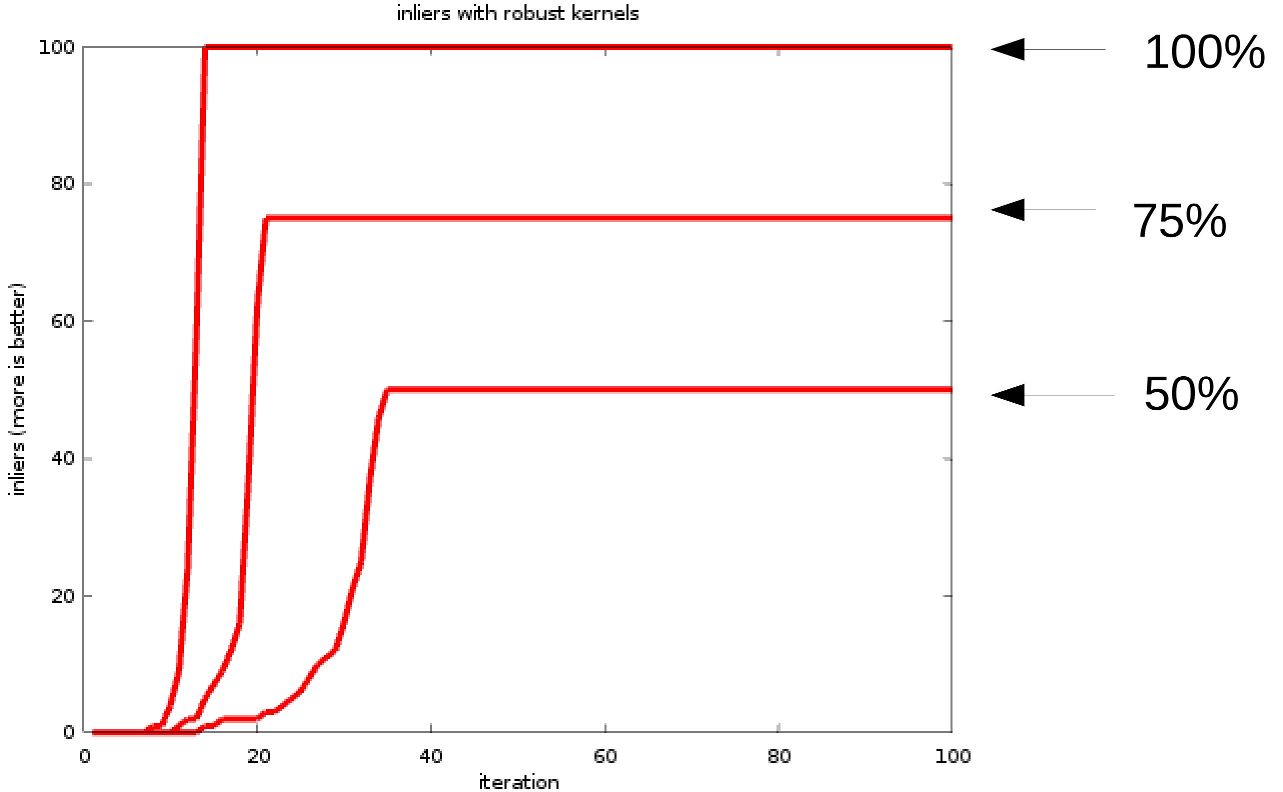
MICP with Outliers: Code

```
function [X, chi_stats, num_inliers]=doICPManifold(X_guess, P, Z, num_iterations, damping, kernel_threshold)
  X=X_guess;
  chi_stats=zeros(1,num_iterations);
  num_inliers=zeros(1,num_iterations);
  for (iteration=1:num_iterations)
    H=zeros(6,6);
    b=zeros(6,1);
    chi_stats(iteration)=0;
    for (i=1:size(P,2))
      [e,J] = errorAndJacobianManifold(X, P(:,i), Z(:,i));
      chi=e'*e;
      if (chi>kernel_threshold)
        e*=sqrt(kernel_threshold/chi);
        chi=kernel_threshold;
      else
        num_inliers(iteration)++;
      endif;
      chi_stats(iteration)+=chi;
      H+=J'*J;
      b+=J'*e;
    endfor
    H+=eye(6)*damping;
    dx=-H\b;
    X=v2t(dx)*X;
  endfor
endfunction
```

Behavior with Outliers

Instead of measuring the F(x) we measure the number of inliers as the algorithm evolves

The closer is the estimated # of inliers to the true fraction the better is our system



Conclusions

- Using manifolds does not necessarily make the derivation more complex
- The convergence is usually improved
- Using robust kernels might help in case of outliers at the cost of lower convergence speed and smaller basin of attraction
- Generate the plots in these slides using the accompanied octave scripts and see what happens when altering the noise
- Follow the methodology!!!!