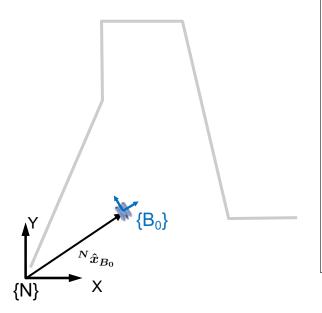


## **Spatial Relationships**

$$egin{aligned} {}^{N}\hat{x}_{k} = egin{bmatrix} {}^{N}\hat{x}_{B_{0}} \ {}^{N}\hat{x}_{B_{1}} \ {}^{:} \ {}^{:} \ {}^{N}P_{k} = egin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}B_{1}} & \ldots & {}^{N}P_{B_{0}B_{k}} \ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & \ldots & {}^{N}P_{B_{1}B_{k}} \ {}^{:} \ {}^{:} \ {}^{:} \ {}^{:} \ {}^{:} \ {}^{:} \ {}^{:} \ {}^{N}P_{B_{k}B_{0}} & {}^{N}P_{B_{k}B_{1}} & \ldots & {}^{N}P_{B_{k}} \ {}^{I} \ {}^$$

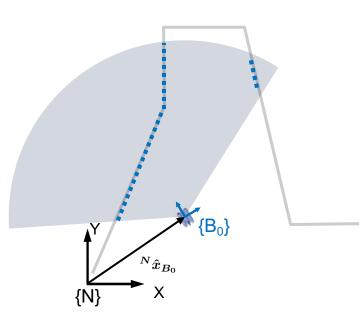
$$\mathcal{M} = egin{bmatrix} B_0 & B_1 S_1 & \dots & B_k S_k \end{bmatrix} \ egin{bmatrix} egin{bmatrix} eta_k & egin{bmatrix} eta_k & egin{bmatrix} eta_k & egin{bmatrix} eta_k & eta_k$$

- > A map is represented by:
  - The robot poses referenced to the N-Frame
  - > The covariances of the robot poses
  - > The scans gathered from the poses
- > The map is built incrementally

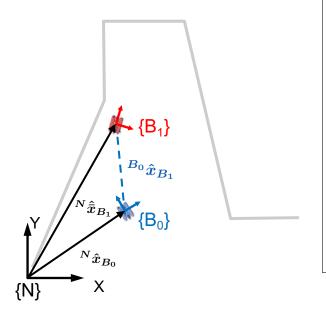


Method Localization  $(\hat{x}_0, P_0)$ 

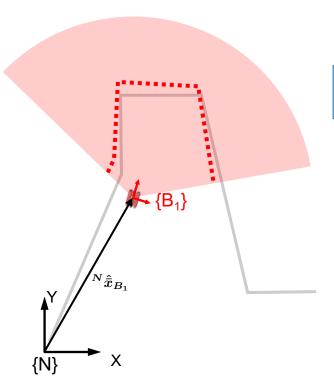
```
[{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];[{}^{B_{0}}S_{0}, {}^{B_{0}}R_{S_{0}}] = GetScan();
                                                                                                                                                                             // Initialize SLAM state vector
[{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                               // Grow the state vector
\mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                    // Store the scan in the map
for k = 1 to steps do
          [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                          // Get input to the motion model
          [N\hat{\hat{x}}_k, N\hat{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
         [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                          // Read navigation sensors
         oldsymbol{z_{p}} = [\ ]; oldsymbol{z_{p}} = [\ ]; \mathcal{H}_{p} = [\ ];
         if ScanAvailable then
                    \begin{split} [^{\boldsymbol{B}_{\boldsymbol{k}}}\boldsymbol{S}_{\boldsymbol{k}}, ^{\boldsymbol{B}_{\boldsymbol{k}}}\boldsymbol{R}_{\boldsymbol{S}_{\boldsymbol{k}}}] &= GetScan(); \\ [^{\boldsymbol{N}}\boldsymbol{\hat{\boldsymbol{x}}}_{\boldsymbol{k}}, ^{\boldsymbol{N}}\boldsymbol{\bar{P}}_{\boldsymbol{k}}] &= AddNewPose(^{\boldsymbol{N}}\boldsymbol{\hat{\boldsymbol{x}}}_{\boldsymbol{k}}, ^{\boldsymbol{N}}\boldsymbol{\bar{P}}_{\boldsymbol{k}}); \end{split} 
                                                                                                                                                                                               // Grow the state vector
                  \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                    // Store the scan in the map
                  \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                        // Get pairs of overlapping scans
                  c = 1; \mathcal{H}_p = [\ ];
                  for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                           // for all overlaps
                            j = \mathcal{H}_o[i];
                                                                                                                                                           // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                           \begin{bmatrix} B_{j}S_{j}, B_{j}R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; \qquad \text{// Get the partial formula} 
 \begin{bmatrix} B_{j}X_{B_{k}} & \in (\ominus^{N}x_{B_{j}}) \oplus^{N}x_{B_{k}}; & \text{//} \\ B_{j}R_{B_{k}} & = J_{1}\oplus J_{\ominus}^{N}P_{B_{j}}J_{\ominus}^{T}J_{1}^{T} + J_{2}\oplus^{N}P_{B_{k}}J_{2}^{T} \\ [z_{r},R_{r}] & = Register(^{B_{j}}S_{j},^{B_{k}}S_{k},^{B_{j}}x_{B_{k}}); & \text{// Scal} \\ \text{if } IndividuallyCompatible(^{B_{j}}x_{B_{k}},^{B_{j}}P_{B_{k}},z_{r},R_{r},\alpha) \text{ then } 
                                                                                                                                                                                 // Get the scans from the map
                                                                                                                                                                                         // Scan Displacement guess
                                                                                                                                                                           // Scan displacement mean & cov
                                     oldsymbol{z_p}[c] = oldsymbol{z_r}; \ oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                                               // Accepted registration
                                \mathcal{H}_p[c] = i; c = c+1;
          \begin{aligned} & [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{N}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ & [{}^{N}\boldsymbol{\hat{x}_k}, {}^{N}\boldsymbol{P_k}] = Update({}^{N}\boldsymbol{\hat{\bar{x}}_k}, {}^{N}\boldsymbol{\bar{P_k}}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \mathcal{H}_p); \end{aligned} 
\mathcal{M}_0 = \lceil
```



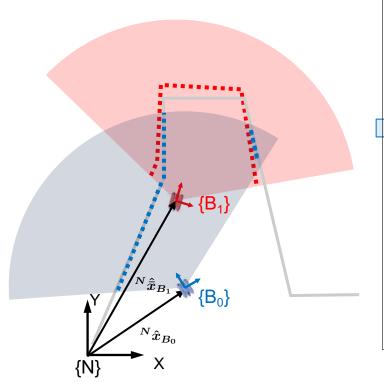
```
Method Localization (\hat{x}_0, P_0)
          [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];[{}^{B_{0}}S_{0}, {}^{B_{0}}R_{S_{0}}] = GetScan();
                                                                                                                                                                                                // Initialize SLAM state vector
          [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                   // Grow the state vector
         \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                       // Store the scan in the map
         for k = 1 to steps do
                    [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                             // Get input to the motion model
                    [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                   [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                             // Read navigation sensors
                   oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                   if ScanAvailable then
                              \begin{split} [^{\boldsymbol{B_k}}\boldsymbol{S_k},^{\boldsymbol{B_k}}\boldsymbol{R_{\boldsymbol{S_k}}}] &= GetScan(); \\ [^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}_k}},^{\boldsymbol{N}}\bar{\boldsymbol{P_k}}] &= AddNewPose(^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}_k}},^{\boldsymbol{N}}\bar{\boldsymbol{P_k}}); \end{split} 
                                                                                                                                                                                                                  // Grow the state vector
                             \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                       // Store the scan in the map
                            \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                                          // Get pairs of overlapping scans
                            c = 1; \mathcal{H}_p = [\ ];
                             for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                // for all overlaps
                                      j = \mathcal{H}_o[i];
                                                                                                                                                                             // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                      \begin{bmatrix} B_{j}S_{j}, B_{j}R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; \qquad \text{// Get the partial formula} 
 \begin{bmatrix} B_{j}X_{B_{k}} & \in (\ominus^{N}x_{B_{j}}) \oplus^{N}x_{B_{k}}; & \text{//} \\ B_{j}R_{B_{k}} & = J_{1}\oplus J_{\ominus}^{N}P_{B_{j}}J_{\ominus}^{T}J_{1}^{T} + J_{2}\oplus^{N}P_{B_{k}}J_{2}^{T} \\ [z_{r},R_{r}] & = Register(^{B_{j}}S_{j},^{B_{k}}S_{k},^{B_{j}}x_{B_{k}}); & \text{// Scal} \\ \text{if } IndividuallyCompatible(^{B_{j}}x_{B_{k}},^{B_{j}}P_{B_{k}},z_{r},R_{r},\alpha) \text{ then } 
                                                                                                                                                                                                    // Get the scans from the map
                                                                                                                                                                                                             // Scan Displacement guess
                                                                                                                                                                                             // Scan displacement mean & cov
                                          \begin{bmatrix} \mathbf{z_p}[c] = \mathbf{z_r}; \ \mathbf{R_p}[c] = \mathbf{R_r}; \\ \mathcal{H}_p[c] = i; \ c = c + 1; \end{bmatrix}
                                                                                                                                                                                                                  // Accepted registration
                    \begin{aligned} [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= ObservationMatrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ [{}^{\boldsymbol{N}}\boldsymbol{\hat{x}_k}, {}^{\boldsymbol{N}}\boldsymbol{P_k}] &= Update({}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, {}^{\boldsymbol{N}}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \boldsymbol{\mathcal{H}_p}); \end{aligned} 
          {}^{N}\hat{ar{x}}_{0} = egin{bmatrix} {}^{N}\hat{x}_{B_{0}} \ {}^{N}\hat{x}_{B_{0}} \end{bmatrix} \; ; \; {}^{N}ar{P}_{0} = egin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}} \ {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}} \end{bmatrix}
             \mathcal{M}_0 = [ [^{B_0}S_0, ^{B_0}R_{S_0}] ]
```



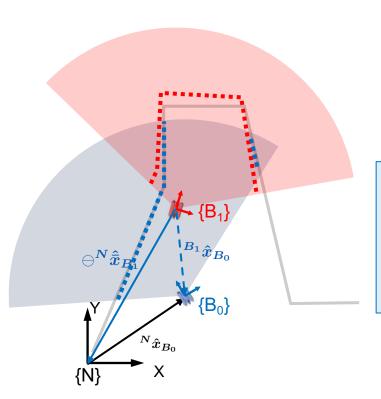
```
Method Localization (\hat{x}_0, P_0)
                [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                                                                                                                                                             // Initialize SLAM state vector
               [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                                                                                                                                                          // Read the Scan from the sensor
              [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                                                                           // Grow the state vector
              \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                                                                         // Store the scan in the map
             for k = 1 to steps do
                                [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                                                                         // Get input to the motion model
                               [N\hat{\bar{x}}_k, N\bar{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
                              [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                                                                                  // Read navigation sensors
                             oldsymbol{z_{oldsymbol{v}}} = [\ ]; oldsymbol{z_{oldsymbol{v}}} = [\ ]; oldsymbol{\mathcal{H}_p} = [\ ];
                             if ScanAvailable then
                                              \begin{aligned} [^{\boldsymbol{B_k}}\boldsymbol{S_k},^{\boldsymbol{B_k}}\boldsymbol{R_{\boldsymbol{S_k}}}] &= GetScan(); \\ [^{\boldsymbol{N}}\boldsymbol{\hat{\boldsymbol{x}}_k},^{\boldsymbol{N}}\bar{\boldsymbol{P}_k}] &= AddNewPose(^{\boldsymbol{N}}\boldsymbol{\hat{\boldsymbol{x}}_k},^{\boldsymbol{N}}\bar{\boldsymbol{P}_k}); \end{aligned} 
                                                                                                                                                                                                                                                                                                                                           // Grow the state vector
                                             \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                                                                                                                                         // Store the scan in the map
                                            \mathcal{H}_{o} = OverlappingScans(^{N}\hat{\bar{x}}_{k}, \mathcal{M});
                                                                                                                                                                                                                                                                                                    // Get pairs of overlapping scans
                                            c = 1; \mathcal{H}_p = [\ ];
                                             for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                                                                                                                                                // for all overlaps
                                                            j = \mathcal{H}_o[i];
                                                                                                                                                                                                                                                                                // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                                           \begin{bmatrix} B_{j}S_{j}, B_{j}R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; \qquad \text{// Get the partial formula} 
 \begin{bmatrix} B_{j}X_{B_{k}} & \in (\ominus^{N}x_{B_{j}}) \oplus^{N}x_{B_{k}}; & \text{//} \\ B_{j}R_{B_{k}} & = J_{1}\oplus J_{\ominus}^{N}P_{B_{j}}J_{\ominus}^{T}J_{1}^{T} + J_{2}\oplus^{N}P_{B_{k}}J_{2}^{T} \\ [z_{r},R_{r}] & = Register(^{B_{j}}S_{j},^{B_{k}}S_{k},^{B_{j}}x_{B_{k}}); & \text{// Scal} \\ \text{if } IndividuallyCompatible(^{B_{j}}x_{B_{k}},^{B_{j}}P_{B_{k}},z_{r},R_{r},\alpha) \text{ then } 
                                                                                                                                                                                                                                                                                                                    // Get the scans from the map
                                                                                                                                                                                                                                                                                                                                  // Scan Displacement guess
                                                                                                                                                                                                                                                                                                          // Scan displacement mean & cov
                                                                           oldsymbol{z_p}[c] = oldsymbol{z_r}; \ oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                                                                                                                                                                                           // Accepted registration
                                                                   \mathcal{H}_p[c] = i; c = c+1;
                              \begin{aligned} [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= ObservationMatrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ [{}^{\boldsymbol{N}}\boldsymbol{\hat{x}_k}, {}^{\boldsymbol{N}}\boldsymbol{P_k}] &= Update({}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, {}^{\boldsymbol{N}}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \boldsymbol{\mathcal{H}_p}); \end{aligned} 
              {}^{N}\hat{ar{x}}_{1} = \left| egin{array}{c} {}^{N}\hat{ar{x}}_{B_{0}} \\ {}^{N}\hat{ar{x}}_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}ar{P}_{1} = \left| egin{array}{c} {}^{N}P_{B_{0}} & {}^{N}ar{P}_{B_{0}B_{1}} \\ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}ar{P}_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}ar{P}_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}ar{P}_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}ar{P}_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} \end{array} 
ight| \;\; ; \;\; {}^{N}P_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}B_{0}} \end{array} 
ight| \;\; ; \;\; {}^{N}P_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}B_{0}} \end{array} 
ight| \;\; ; \;\; {}^{N}P_{1} = \left| egin{array}{c} {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}B_{0}} \end{array} 
ight| \;\; ; \;\; {}^{N}P_{1} = \left| egin{array}{c} {}^{N}P_{1} = \left|
                    \mathcal{M}_1 = \lceil \lceil ^{B_0}S_0, ^{B_0}R_{S_0} 
ceil
                                                                                                                                                                                                                                                                                                                             oldsymbol{z_m} R_m
```



```
Method Localization (\hat{x}_0, P_0)
               [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                                                                                                                                    // Initialize SLAM state vector
              [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                                                                                                                                 // Read the Scan from the sensor
             [^{N}\hat{\bar{x}}_{0}, ^{N}P_{0}] = AddNewPose(^{N}\hat{x}_{0}, ^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                                                 // Grow the state vector
             \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                                               // Store the scan in the map
             for k = 1 to steps do
                             [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                                                // Get input to the motion model
                            [N\hat{\bar{x}}_k, N\bar{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
                            [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                                                        // Read navigation sensors
                            \boldsymbol{z_p} = [\ ]; \boldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                          if ScanAvailable then
                                           [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                                          [^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k}] = AddNewPose(^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k});
                                                                                                                                                                                                                                                                                                               // Grow the state vector
                                         \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                                                                                                               // Store the scan in the map
                                         \mathcal{H}_{o} = OverlappingScans(^{\mathbf{N}}\hat{\mathbf{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                                                                                                                           // Get pairs of overlapping scans
                                         c = 1; \mathcal{H}_p = [\ ];
                                         for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                                                                                                                  // for all overlaps
                                                       j = \mathcal{H}_o[i];
                                                                                                                                                                                                                                                         // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                                     \begin{bmatrix} B_{j}S_{j}, B_{j}R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of } I = I_{0}[i]; \qquad \text{// Get the partial of }
                                                                                                                                                                                                                                                                                           // Get the scans from the map
                                                                                                                                                                                                                                                                                                       // Scan Displacement guess
                                                                                                                                                                                                                                                                                // Scan displacement mean & cov
                                                            \begin{bmatrix} \mathbf{z_p}[c] = \mathbf{z_r}; \ \mathbf{R_p}[c] = \mathbf{R_r}; \\ \mathcal{H}_p[c] = i; \ c = c + 1; \end{bmatrix}
                                                                                                                                                                                                                                                                                                               // Accepted registration
                           \begin{aligned} & [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{N}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ & [{}^{N}\boldsymbol{\hat{x}_k}, {}^{N}\boldsymbol{P_k}] = Update({}^{N}\boldsymbol{\hat{\bar{x}}_k}, {}^{N}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \mathcal{H}_p); \end{aligned} 
            {}^{N}\hat{ar{x}}_{1} = egin{bmatrix} {}^{N}\hat{ar{x}}_{B_{0}} \\ {}^{N}\hat{ar{x}}_{B_{1}} \\ {}^{N}\hat{ar{x}}_{B_{1}} \end{bmatrix} \; ; \; {}^{N}ar{P}_{1} = egin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}ar{P}_{B_{0}B_{1}} & {}^{N}ar{P}_{B_{0}B_{1}} \\ {}^{N}ar{P}_{B_{1}B_{0}} & {}^{N}ar{P}_{B_{1}} & {}^{N}ar{P}_{B_{1}} \\ {}^{N}ar{P}_{B_{1}B_{0}} & {}^{N}ar{P}_{B_{1}} & {}^{N}ar{P}_{B_{1}} \end{bmatrix}
                   \mathcal{M}_1 = [[B_0 S_0, {}^{B0}R_{S_0}]]^{[B_1}S_1, {}^{B1}R_{S_1}]
```

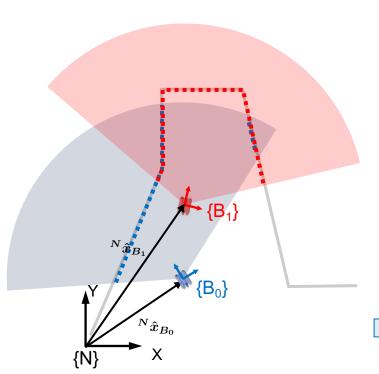


```
Method Localization (\hat{x}_0, P_0)
         [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                              // Initialize SLAM state vector
         [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                            // Read the Scan from the sensor
        [^{N}\hat{\bar{x}}_{0}, ^{N}P_{0}] = AddNewPose(^{N}\hat{x}_{0}, ^{N}\bar{P}_{0});
                                                                                                                                                                                                // Grow the state vector
        \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                     // Store the scan in the map
        for k = 1 to steps do
                  [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                            // Get input to the motion model
                  [N\hat{\bar{x}}_k, N\bar{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
                 [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                          // Read navigation sensors
                 \boldsymbol{z_p} = [\ ]; \boldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                 if ScanAvailable then
                           [^{\boldsymbol{B_k}}\boldsymbol{S_k}, ^{\boldsymbol{B_k}}\boldsymbol{R_{S_k}}] = GetScan();
                          [N\hat{\bar{x}}_{k}, N\bar{P}_{k}] = AddNewPose(N\hat{\bar{x}}_{k}, N\bar{P}_{k});
                                                                                                                                                                                               // Grow the state vector
                         \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
\mathcal{H}_o = OverlappingScans(^{N} \hat{\bar{x}}_k, \mathcal{M});
                                                                                                                                                                                     // Store the scan in the map
                                                                                                                                                                         // Get pairs of overlapping scans
                          c = 1; \mathcal{H}_p = [\ ];
                          for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                           // for all overlaps
                                   j = \mathcal{H}_o[i];
                                                                                                                                                             // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                  \begin{bmatrix} B_{j}S_{j}, B_{j}R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; \qquad \text{// Get the partial formula} 
 \begin{bmatrix} B_{j}X_{B_{k}} & \in (\ominus^{N}x_{B_{j}}) \oplus^{N}x_{B_{k}}; & \text{//} \\ B_{j}R_{B_{k}} & = J_{1}\oplus J_{\ominus}^{N}P_{B_{j}}J_{\ominus}^{T}J_{1}^{T} + J_{2}\oplus^{N}P_{B_{k}}J_{2}^{T} \\ [z_{r},R_{r}] & = Register(^{B_{j}}S_{j},^{B_{k}}S_{k},^{B_{j}}x_{B_{k}}); & \text{// Scal} \\ \text{if } IndividuallyCompatible(^{B_{j}}x_{B_{k}},^{B_{j}}P_{B_{k}},z_{r},R_{r},\alpha) \text{ then } 
                                                                                                                                                                                  // Get the scans from the map
                                                                                                                                                                                          // Scan Displacement guess
                                                                                                                                                                        // Scan displacement mean & cov
                                           oldsymbol{z_p}[c] = oldsymbol{z_r}; \ oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                                               // Accepted registration
                                      \mathcal{H}_p[c] = i; c = c + 1;
                  \begin{aligned} [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= ObservationMatrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ [{}^{\boldsymbol{N}}\boldsymbol{\hat{x}_k}, {}^{\boldsymbol{N}}\boldsymbol{P_k}] &= Update({}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, {}^{\boldsymbol{N}}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \boldsymbol{\mathcal{H}_p}); \end{aligned} 
        egin{aligned} {}^{N}\hat{x}_{1} = egin{bmatrix} {}^{N}\hat{x}_{B_{0}} \ {}^{N}\hat{x}_{B_{1}} \ {}^{N}\hat{x}_{B_{1}} \end{bmatrix} \; ; \; {}^{N}P_{1} = egin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}B_{1}} & {}^{N}P_{B_{0}B_{1}} \ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & {}^{N}P_{B_{1}} \ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & {}^{N}P_{B_{1}} \ \end{array} \; 
            \mathcal{M}_1 = \left[\left[egin{array}{cccc} B_0 S_0, B^0 R_{S_0} 
ight] \left[egin{array}{cccc} B_1 S_1, B^1 R_{S_1} 
ight] \end{array}
ight] \quad \left[egin{array}{cccc} z_m & R_m 
ight] 
ight]
             \mathcal{H}_{n} = \begin{bmatrix} 0 \end{bmatrix}
```



```
Method Localization (\hat{x}_0, P_0)
             [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                 // Initialize SLAM state vector
             [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                               // Read the Scan from the sensor
            [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                 // Grow the state vector
            \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                       // Store the scan in the map
            for k = 1 to steps do
                     [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                               // Get input to the motion model
                     [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                    [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                            // Read navigation sensors
                    oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                    if ScanAvailable then
                             [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                            [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = AddNewPose({}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k});
                                                                                                                                                                                // Grow the state vector
                            \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                       // Store the scan in the map
                            \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M}):
                                                                                                                                                            // Get pairs of overlapping scans
                            c=1; \mathcal{H}_p=[\ ];
                            for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                           // for all overlaps
                                   j = \mathcal{H}_o[i];
                                                                                                                                                 // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                    \begin{bmatrix} B_j S_j, B_j R_{S_j} \end{bmatrix} = \mathcal{M}[j]; 
 \begin{bmatrix} B_j X_{B_k} = (\ominus^N x_{B_j}) \oplus^N x_{B_k} ; \\ B_j P_{B_k} = J_{1 \oplus} J_{\ominus}^N P_{B_j} J_{\ominus}^T J_{1 \oplus}^T + J_{2 \oplus}^N P_{B_k} J_{2 \oplus}^T \end{bmatrix} 
                                                                                                                                                                    // Get the scans from the map
                                                                                                                                                                           // Scan Displacement guess
                                      [\boldsymbol{z_r}, \boldsymbol{R_r}] = Register(\check{B_j} \boldsymbol{S_i}, \check{B_k} \boldsymbol{S_k}, \check{B_j} \boldsymbol{x_{B_k}});
                                                                                                                                                            // Scan displacement mean & cov
                                    if IndividuallyCompatible(^{B_j}x_{B_k},^{B_j}P_{B_k},z_r,R_r,\alpha) then
                                           \boldsymbol{z_p}[c] = \boldsymbol{z_r}; \ \boldsymbol{R_p}[c] = \boldsymbol{R_r};
                                                                                                                                                                                // Accepted registration
                                       \mathcal{H}_n[c] = i; c = c + 1;
                    \begin{aligned} [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= ObservationMatrix(\mathcal{H}_p, {}^{N}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ [{}^{N}\boldsymbol{\hat{x}_k}, {}^{N}\boldsymbol{P_k}] &= Update({}^{N}\boldsymbol{\hat{\bar{x}}_k}, {}^{N}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \boldsymbol{\mathcal{H}_p}); \end{aligned} 
           {}^{N}\hat{x}_{1} = egin{bmatrix} {}^{N}\hat{x}_{B_{0}} \ {}^{N}\hat{x}_{B_{1}} \ {}^{N}\hat{x}_{B_{1}} \end{bmatrix} \; ; \; {}^{N}P_{1} = egin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}B_{1}} & {}^{N}P_{B_{0}B_{1}} \ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & {}^{N}P_{B_{1}} \ {}^{N}P_{B_{1}} & {}^{N}P_{B_{1}} \end{bmatrix} \; ,
               \mathcal{M}_1 = \left[\left[egin{aligned} B_0 S_0, B^0 R_{S_0} 
ight]\left[egin{aligned} B_1 S_1, B^1 R_{S_1} 
ight] \end{aligned}
ight]
     \mathcal{H}_p = \begin{bmatrix} 0 \\ \end{bmatrix}
B_1 x_{B_0} = (\ominus^N x_{B_2}) \oplus {}^N x_{B_0})
[z_{n_1}, R_{n_1}] = Register(^{B_1}S_1, ^{B_0}S_0, ^{B_2}x_{B_0})
                 oldsymbol{z_p} = egin{bmatrix} oldsymbol{z_{p_1}} \end{bmatrix} \; ; \; oldsymbol{R_p} = egin{bmatrix} oldsymbol{R_{p_1}} \end{bmatrix}
```

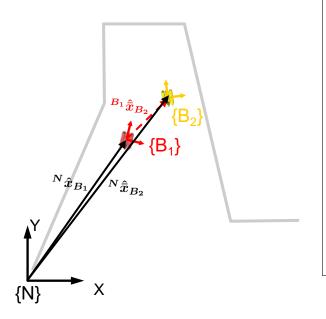
```
Method Localization (\hat{x}_0, P_0)
                                                                                                                                                                                                                                    [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   // Initialize SLAM state vector
                                                                                                                                                                                                                                   [B_0S_0, B_0R_{S_0}] = GetScan();
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               // Read the Scan from the sensor
                                                                                                                                                                                                                                  [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       // Grow the state vector
                                                                                                                                                                                                                                  \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 // Store the scan in the map
                                                                                                                                                                                                                                  for k = 1 to steps do
                                                                                                                                                                                                                                                      [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              // Get input to the motion model
                                                                                                                                                                                                                                                    [N\hat{\bar{x}}_k, N\hat{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
                                                                                                                                                                                                                                                     [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            // Read navigation sensors
                                                                                                                                                                               Method ObservationMatrix (\mathcal{H}_n, {}^N\hat{\bar{x}}_k, z_m, R_m, z_n, R_n)
                                                                                                                                                                                                 H_p=0;
                                                                                                                                                                                                for i = 1 to length(z_p) do
                                                                                                                                                                                                      | H_p[i,1] = J_{1\oplus}(\ominus^N \hat{\bar{x}}_{B_k}, {}^N \hat{x}_k[\mathcal{H}_p[i]]) J_{\ominus}({}^N \hat{\bar{x}}_{B_k}); 
 | H_p[i,\mathcal{H}_p[i]] = J_{2\oplus}({}^N \hat{\bar{x}}_{B_k}); 
                                                                                                                                                                                              egin{aligned} egin{aligned\\ egin{aligned} egi
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       // Indep. measure&displacement noise
                                                                                                                                                                                            egin{aligned} oldsymbol{H_k} = egin{bmatrix} oldsymbol{J_{h_x}(\hat{ar{x}_k})} \ oldsymbol{H_p} \end{bmatrix}; \end{aligned}
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  // Stack measurements and Displ. observ.
                                                - {B₁}
                                                                                                                                                                                                                                                                                                          \mathcal{H}_{p}[c] = i; c = c + 1;
                                           \{B_0\}
                                                                                                                                                                                                                                                [z_k, R_k, H_k, V_k] = ObservationMatrix(\mathcal{H}_p, {}^{N}\hat{\bar{x}}_k, z_m, R_m, z_p, R_p);[{}^{N}\hat{x}_k, {}^{N}P_k] = Update({}^{N}\hat{\bar{x}}_k, {}^{N}\bar{P}_k, z_k, R_k, H_k, V_k, \mathcal{H}_p);
^{N}\hat{x}_{B_{0}}
                                                                                                                                                                                                                                  egin{aligned} N \hat{x}_1 = egin{bmatrix} N \hat{x}_{B_0} \ N \hat{x}_{B_1} \ N \hat{x}_{B_1} \end{bmatrix} \; ; \; N P_1 = egin{bmatrix} N P_{B_0} & N P_{B_0} B_1 & N P_{B_0} B_1 \ N P_{B_1} B_0 & N P_{B_1} & N P_{B_1} \ N P_{B_1} B_0 & N P_{B_1} & N P_{B_1} \end{bmatrix} \end{aligned}
                                                                                                                                                                                                                                          \mathcal{M}_1 = \left[ \left[ {}^{B_0}S_0, {}^{B0}R_{S_0} \right] \left[ {}^{B_1}S_1, {}^{B1}R_{S_1} \right] \right]
                                                                                                                                                                                                                                            \mathcal{H}_n = \begin{bmatrix} 0 \end{bmatrix}
                                                                                                                                                                                                     egin{aligned} egin{aligned\\ egin{aligned} egi
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                H_k = egin{bmatrix} H_m & 0 \ 0 & H_n \end{bmatrix} \; ; \; V_k = egin{bmatrix} I & 0 \ 0 & I \end{bmatrix}
                                                                                                                                                                                                                                                 oldsymbol{z_n} = egin{bmatrix} oldsymbol{z_{n_1}} \end{bmatrix} \; ; \; oldsymbol{R_n} = egin{bmatrix} oldsymbol{R_{n_1}} \end{bmatrix}
```



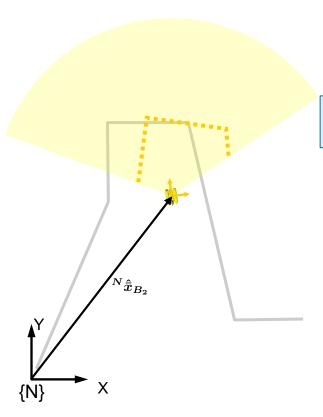
```
Method Localization (\hat{x}_0, P_0)
        [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                 // Initialize SLAM state vector
       [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                               // Read the Scan from the sensor
       [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                // Grow the state vector
      \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                       // Store the scan in the map
      for k = 1 to steps do
               [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                               // Get input to the motion model
               [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = Prediction({}^{N}\hat{x}_{k-1}, {}^{N}P_{k-1}, u_{k}, Q_{k});
              [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                           // Read navigation sensors
              oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
              if ScanAvailable then
                      [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                      [N\hat{\bar{x}}_{k}, N\bar{P}_{k}] = AddNewPose(N\hat{\bar{x}}_{k}, N\bar{P}_{k});
                                                                                                                                                               // Grow the state vector
                      \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                       // Store the scan in the map
                     \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                             // Get pairs of overlapping scans
                     c = 1; \mathcal{H}_p = [\ ];
                      for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                         // for all overlaps
                            j = \mathcal{H}_o[i];
                                                                                                                                   // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                             \begin{bmatrix} B_{j} S_{j}, B_{j} R_{S_{i}} \end{bmatrix} = \mathcal{M}[j]; 
 \begin{bmatrix} B_{j} S_{j}, B_{j} R_{S_{i}} \end{bmatrix} = \mathcal{M}[j]; 
 B_{j} x_{B_{k}} = (\ominus^{N} x_{B_{j}}) \oplus^{N} x_{B_{k}}; 
 B_{j} P_{B_{k}} = J_{1} \oplus J_{\ominus}^{N} P_{B_{\underline{j}}} J_{\ominus}^{T} J_{1} \oplus^{T} + J_{2} \oplus^{N} P_{B_{k}} J_{2} \oplus^{T} 
                                                                                                                                                    // Get the scans from the map
                                                                                                                                                           // Scan Displacement guess
                               [\boldsymbol{z_r}, \boldsymbol{R_r}] = Register(\overset{\boldsymbol{\beta_j}}{B_i} \boldsymbol{S_j}, \overset{\boldsymbol{\beta_k}}{B_k} \boldsymbol{S_k}, \overset{\boldsymbol{\beta_j}}{B_j} \boldsymbol{x_{B_L}});
                                                                                                                                                // Scan displacement mean & cov
                             if IndividuallyCompatible(^{B_j}x_{B_k},^{B_j}P_{B_k},z_r,R_r,\alpha) then
                                    \boldsymbol{z_p}[c] = \boldsymbol{z_r}; \ \boldsymbol{R_p}[c] = \boldsymbol{R_r};
                                                                                                                                                               // Accepted registration
                                \mathcal{H}_{n}[c] = i; c = c + 1;
               [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p});
               [{}^{N}\hat{\boldsymbol{x}}_{k}, {}^{N}\boldsymbol{P}_{k}] = Update({}^{N}\hat{\boldsymbol{x}}_{k}, {}^{N}\boldsymbol{\bar{P}}_{k}, \boldsymbol{z}_{k}, \boldsymbol{R}_{k}, \boldsymbol{H}_{k}, \boldsymbol{V}_{k}, \mathcal{H}_{D});
```

$$egin{aligned} {}^{N}\hat{x}_{1} = egin{bmatrix} N \hat{x}_{B_{0}} \ N \hat{x}_{B_{1}} \ N \hat{x}_{B_{1}} \end{bmatrix} \; ; \; {}^{N}P_{1} = egin{bmatrix} N P_{B_{0}} & N P_{B_{0}B_{1}} & N P_{B_{0}B_{1}} \ N P_{B_{1}B_{0}} & N P_{B_{1}} & N P_{B_{1}} \ N P_{B_{1}B_{0}} & N P_{B_{1}} & N P_{B_{1}} \end{bmatrix} \end{aligned}$$

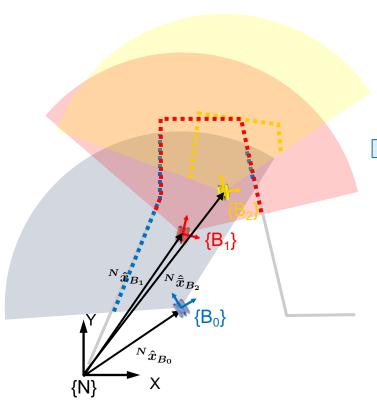
$$\mathcal{M}_1 = \left[ \begin{bmatrix} B_0 S_0, B^0 R_{S_0} \end{bmatrix} \begin{bmatrix} B_1 S_1, B^1 R_{S_1} \end{bmatrix} \right] \quad \left[ oldsymbol{z_m} \quad oldsymbol{R_m} 
ight]$$



```
Method Localization (\hat{x}_0, P_0)
         [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                              // Initialize SLAM state vector
        [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                            // Read the Scan from the sensor
       [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                              // Grow the state vector
       \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                    // Store the scan in the map
       for k = 1 to steps do
                [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                            // Get input to the motion model
                [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = Prediction({}^{N}\hat{x}_{k-1}, {}^{N}P_{k-1}, u_{k}, Q_{k});
               [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                         // Read navigation sensors
               oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
               if ScanAvailable then
                        [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                        [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = AddNewPose({}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k});
                                                                                                                                                                              // Grow the state vector
                        \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                    // Store the scan in the map
                       \mathcal{H}_{o} = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                         // Get pairs of overlapping scans
                       c = 1; \mathcal{H}_{p} = [\ ];
                        for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                         // for all overlaps
                               j = \mathcal{H}_o[i];
                                                                                                                                              // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                               \begin{aligned} &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}x_{B_k} = (\ominus^N x_{B_j}) \oplus^N x_{B_k}; \\ &[^{B_j}P_{B_k} = J_1 \oplus J_{\ominus}^N P_{B_j}J_{\ominus}^T J_{1\oplus}^T + J_{2\oplus}^N P_{B_k}J_{2\oplus}^T \\ &[z_r, R_r] = Register(^{B_j}S_j,^{B_k}S_k,^{B_j}x_{B_k}); \end{aligned} 
                                                                                                                                                                  // Get the scans from the map
                                                                                                                                                                         // Scan Displacement guess
                                                                                                                                                            // Scan displacement mean & cov
                               if IndividuallyCompatible(^{B_j}x_{B_k}, ^{B_j}P_{B_k}, z_r, R_r, \alpha) then
                                       oldsymbol{z_p}[c] = oldsymbol{z_r}; oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                              // Accepted registration
                                   \mathcal{H}_p[c] = i; c = c+1;
                \begin{aligned} [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= ObservationMatrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ [{}^{\boldsymbol{N}}\boldsymbol{\hat{x}_k}, {}^{\boldsymbol{N}}\boldsymbol{P_k}] &= Update({}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, {}^{\boldsymbol{N}}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \boldsymbol{\mathcal{H}_p}); \end{aligned} 
       egin{aligned} N \, \hat{ar{x}}_2 = egin{bmatrix} ^N \hat{ar{x}}_{B_0} \ ^N \hat{ar{x}}_{B_1} \ ^N \hat{ar{x}}_{B_2} \end{bmatrix} \; ; \; ^N ar{P}_2 = egin{bmatrix} ^N P_{B_0} & ^N P_{B_0} & ^N P_{B_0} B_1 \ ^N P_{B_1 B_0} & ^N P_{B_1} & ^N ar{P}_{B_1 B_2} \ ^N ar{P}_{B_2 B_0} & ^N ar{P}_{B_2 B_1} & ^N ar{P}_{B_2} \end{bmatrix} \end{aligned}
           \mathcal{M}_2 = \left[ \left[ {}^{B_0}S_0, {}^{B0}R_{S_0} \right] \left[ {}^{B_1}S_1, {}^{B1}R_{S_1} \right] \right]
                                                                                                                                                                      z_m R_m
            \mathcal{H}_n = \begin{bmatrix} 0 \end{bmatrix}
```

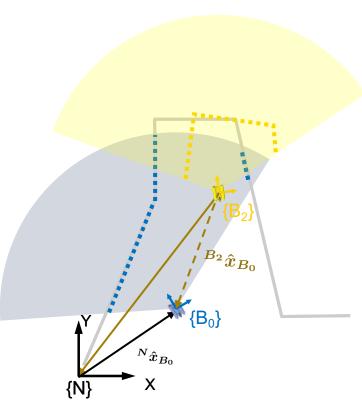


```
Method Localization (\hat{x}_0, P_0)
              [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                                                                                                           // Initialize SLAM state vector
             [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                                                                                                        // Read the Scan from the sensor
             [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                    // Grow the state vector
           \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                     // Store the scan in the map
           for k = 1 to steps do
                          [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                       // Get input to the motion model
                          [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                          [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                            // Read navigation sensors
                         \boldsymbol{z_p} = [\ ]; \boldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                         if ScanAvailable then
                                       [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                                      [^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k}] = AddNewPose(^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k});
                                                                                                                                                                                                                                                                                   // Grow the state vector
                                      \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                                                                                     // Store the scan in the map
                                      \mathcal{H}_{o} = OverlappingScans(^{\mathbf{N}}\hat{\mathbf{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                                                                                                   // Get pairs of overlapping scans
                                     c = 1; \mathcal{H}_p = [\ ];
                                      for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                                                                                     // for all overlaps
                                                  j = \mathcal{H}_o[i];
                                                                                                                                                                                                                                  // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                                 \begin{aligned} &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}x_{B_k} = (\ominus^Nx_{B_j}) \oplus^Nx_{B_k}; \\ &[^{B_j}P_{B_k} = J_1 \oplus J_{\ominus}^N P_{B_j}J_{\ominus}^T J_{1\oplus}^T + J_{2\oplus}^N P_{B_k}J_{2\oplus}^T \\ &[z_r, R_r] = Register(^{B_j}S_j,^{B_k}S_k,^{B_j}x_{B_k}); \end{aligned} 
                                                                                                                                                                                                                                                                // Get the scans from the map
                                                                                                                                                                                                                                                                            // Scan Displacement guess
                                                                                                                                                                                                                                                   // Scan displacement mean & cov
                                                 if Individually Compatible(^{B_j}x_{B_k},^{B_j}P_{B_k},z_r,R_r,\alpha) then
                                                      \begin{bmatrix} \mathbf{z_p}[c] = \mathbf{z_r}; \ \mathbf{R_p}[c] = \mathbf{R_r}; \\ \mathcal{H}_p[c] = i; \ c = c + 1; \end{bmatrix}
                                                                                                                                                                                                                                                                                   // Accepted registration
                         \begin{aligned} & [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{N}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ & [{}^{N}\boldsymbol{\hat{x}_k}, {}^{N}\boldsymbol{P_k}] = Update({}^{N}\boldsymbol{\hat{\bar{x}}_k}, {}^{N}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \mathcal{H}_p); \end{aligned} 
           egin{aligned} N \, \hat{ar{x}}_2 &= egin{bmatrix} N \, \hat{ar{x}}_{B_0} \ N \, \hat{ar{x}}_{B_1} \ N \, \hat{ar{x}}_{B_2} \ N \, \hat{ar{x}}_{B_2} \end{bmatrix} \; ; \; N \, ar{P}_2 &= egin{bmatrix} N \, P_{B_0} & N \, P_{B_0} & N \, P_{B_0B_1} & N \, ar{P}_{B_0B_2} \ N \, P_{B_1B_0} & N \, P_{B_1} & N \, ar{P}_{B_1B_2} \ N \, ar{P}_{B_1B_2} & N \, ar{P}_{B_1B_2} \ N \, ar{P}_{B_2B_0} & N \, ar{P}_{B_2B_1} & N \, ar{P}_{B_2} & N \, ar{P}_{B_2} \ N \, ar{P
                   \mathcal{H}_n = \begin{bmatrix} 0 \end{bmatrix}
```

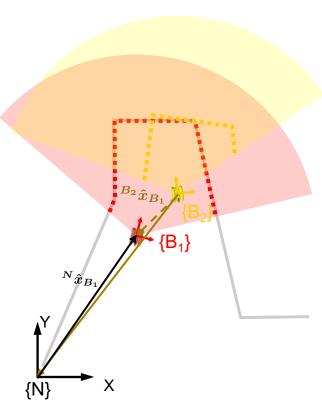


```
Method Localization (\hat{x}_0, P_0)
         [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                        // Initialize SLAM state vector
         [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                      // Read the Scan from the sensor
        [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                         // Grow the state vector
       \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                               // Store the scan in the map
       for k = 1 to steps do
                  [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                      // Get input to the motion model
                 [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                 [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                    // Read navigation sensors
                 oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                 if ScanAvailable then
                          [^{\boldsymbol{B_k}}\boldsymbol{S_k}, ^{\boldsymbol{B_k}}\boldsymbol{R_{S_k}}] = GetScan();
                          [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = AddNewPose({}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k});
                                                                                                                                                                                         // Grow the state vector
                        \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
\mathcal{H}_o = OverlappingScans(^{N} \hat{\bar{x}}_k, \mathcal{M});
                                                                                                                                                                               // Store the scan in the map
                                                                                                                                                                   // Get pairs of overlapping scans
                         c=1; \mathcal{H}_p=[\ ];
                         for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                    // for all overlaps
                                  j = \mathcal{H}_o[i];
                                                                                                                                                        // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                                  \begin{split} &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}x_{B_k} = (\ominus^Nx_{B_j}) \oplus^Nx_{B_k}; \\ &[^{B_j}P_{B_k} = J_1 \oplus J_{\ominus}^N P_{B_j}J_{\ominus}^T J_{1\oplus}^T + J_{2\oplus}^N P_{B_k}J_{2\oplus}^T \\ &[z_r, R_r] = Register(^{B_j}S_j,^{B_k}S_k,^{B_j}x_{B_k}); \end{split} 
                                                                                                                                                                            // Get the scans from the map
                                                                                                                                                                                    // Scan Displacement guess
                                                                                                                                                                     // Scan displacement mean & cov
                                 if Individually Compatible(^{B_j}x_{B_k},^{B_j}P_{B_k},z_r,R_r,\alpha) then
                                          oldsymbol{z_p}[c] = oldsymbol{z_r}; \ oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                                         // Accepted registration
                                     \mathcal{H}_p[c] = i; c = c + 1;
                \begin{split} & [\bar{\boldsymbol{z}_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{N}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\ & [{}^{N}\boldsymbol{\hat{x}_k}, {}^{N}\boldsymbol{P_k}] = Update({}^{N}\boldsymbol{\hat{\bar{x}}_k}, {}^{N}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \mathcal{H}_p); \end{split}
       egin{aligned} N \hat{ar{x}}_2 &= egin{bmatrix} N \hat{ar{x}}_{B_0} \ N \hat{ar{x}}_{B_1} \ N \hat{ar{x}}_{B_2} \ N \hat{ar{x}}_{B_2} \end{bmatrix} \; ; \; N ar{P}_2 &= egin{bmatrix} N P_{B_0} & N P_{B_0} B_1 & N ar{P}_{B_0 B_1} & N ar{P}_{B_0 B_2} \ N P_{B_1 B_0} & N P_{B_1} & N ar{P}_{B_1 B_2} & N ar{P}_{B_1 B_2} \ N ar{P}_{B_2 B_0} & N ar{P}_{B_2 B_1} & N ar{P}_{B_2} & N ar{P}_{B_2} \ N ar{P}_{B_2} & N ar{P}_{B_2} & N ar{P}_{B_2} \ \end{pmatrix} \end{aligned}
           \mathcal{M}_2 = \left[ \begin{bmatrix} B_0 S_0, B^0 R_{S_0} \end{bmatrix} \begin{bmatrix} B_1 S_1, B^1 R_{S_1} \end{bmatrix} \begin{bmatrix} B_2 S_2, B^2 R_{S_2} \end{bmatrix} \right] \quad \begin{bmatrix} \boldsymbol{z_m} & R_m \end{bmatrix}
```

 $\mathcal{H}_{p} = \begin{bmatrix} 0 & 1 \end{bmatrix}$ 



```
Method Localization (\hat{x}_0, P_0)
                                 [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                                                                                                                                                                                                                                                    // Initialize SLAM state vector
                                [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                                                                                                                                                                                                                                                // Read the Scan from the sensor
                               [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                                                                                                                                                         // Grow the state vector
                             \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                                                                                                                                                  // Store the scan in the map
                             for k = 1 to steps do
                                                  [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                                                                                                                                               // Get input to the motion model
                                                  [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                                                  [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                                                                                                                                                             // Read navigation sensors
                                                 \boldsymbol{z_p} = [\ ]; \boldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
                                                 if ScanAvailable then
                                                                     [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                                                                   [^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k}] = AddNewPose(^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k});
                                                                                                                                                                                                                                                                                                                                                                                                                       // Grow the state vector
                                                                  \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                                                                                                                                                                                                                  // Store the scan in the map
                                                                  \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                                                                                                                                                                                                                         // Get pairs of overlapping scans
                                                                  c = 1; \mathcal{H}_p = [\ ];
                                                                  for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                                                                                                                               ^{\prime\prime} for all overlaps ^{\prime\prime} // Get the pair: ^{\prime\prime} ^{\prime\prime} overlaps ^{\prime\prime} ^
                                                                                     j = \mathcal{H}_o[i];
                                                                                    \begin{array}{l} \begin{bmatrix} J - R_{O[i]}, \\ B_j S_j, B_j R_{S_j} \end{bmatrix} = \mathcal{M}[j]; \\ B_j x_{B_k} = (\ominus^N x_{B_j}) \oplus^N x_{B_k}; \\ B_j P_{B_k} = J_{1 \oplus} J_{\ominus}^N P_{B_j} J_{\ominus}^T J_{1 \oplus}^T + J_{2 \oplus}^N P_{B_k} J_{2 \oplus}^T \end{array} 
                                                                                                                                                                                                                                                                                                                                                                                            // Get the scans from the map
                                                                                                                                                                                                                                                                                                                                                                                                            // Scan Displacement guess
                                                                                          [\boldsymbol{z_r}, \boldsymbol{R_r}] = Register(\overset{\boldsymbol{\beta_j}}{B_i} \boldsymbol{S_j}, \overset{\boldsymbol{\beta_k}}{B_k} \boldsymbol{S_k}, \overset{\boldsymbol{\beta_j}}{B_j} \boldsymbol{x_{B_L}});
                                                                                                                                                                                                                                                                                                                                                                             // Scan displacement mean & cov
                                                                                    if IndividuallyCompatible(^{B_j}x_{B_k},^{B_j}P_{B_k},z_r,R_r,\alpha) then
                                                                                                       \boldsymbol{z_n}[c] = \boldsymbol{z_r}; \ \boldsymbol{R_n}[c] = \boldsymbol{R_r};
                                                                                                                                                                                                                                                                                                                                                                                                                       // Accepted registration
                                                                                            \mathcal{H}_{p}[c] = i; c = c + 1;
                                               [\mathbf{z_k}, \mathbf{R_k}, \mathbf{H_k}, \mathbf{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{\mathbf{N}}\mathbf{\hat{\bar{z}}_k}, \mathbf{z_m}, \mathbf{R_m}, \mathbf{z_p}, \mathbf{R_p}); \\ [{}^{\mathbf{N}}\mathbf{\hat{x}_k}, {}^{\mathbf{N}}\mathbf{P_k}] = Update({}^{\mathbf{N}}\mathbf{\hat{z}_k}, {}^{\mathbf{N}}\mathbf{\bar{P}_k}, \mathbf{z_k}, \mathbf{R_k}, \mathbf{H_k}, \mathbf{V_k}, \mathcal{H}_p);
                             {}^{N}\hat{\bar{x}}_{2} = \begin{bmatrix} {}^{N}\hat{x}_{B_{0}} \\ {}^{N}\hat{x}_{B_{1}} \\ {}^{N}\hat{\bar{x}}_{B_{2}} \\ {}^{N}\hat{\bar{x}}_{B_{2}} \end{bmatrix} \; ; \; {}^{N}\bar{P}_{2} = \begin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}B_{1}} & {}^{N}\bar{P}_{B_{0}B_{2}} & {}^{N}\bar{P}_{B_{0}B_{2}} \\ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & {}^{N}\bar{P}_{B_{1}B_{2}} & {}^{N}\bar{P}_{B_{1}B_{2}} \\ {}^{N}\bar{P}_{B_{2}B_{0}} & {}^{N}\bar{P}_{B_{2}B_{1}} & {}^{N}\bar{P}_{B_{2}} & {}^{N}\bar{P}_{B_{2}} \\ {}^{N}\bar{P}_{B_{2}B_{0}} & {}^{N}\bar{P}_{B_{2}B_{1}} & {}^{N}\bar{P}_{B_{2}} & {}^{N}\bar{P}_{B_{2}} \end{bmatrix}
                                     \mathcal{M}_2 = \left[ \left[ \overline{^{B_0}S_0, ^{B0}R_{S_0}} \right] \left[ ^{B_1}S_1, ^{B1}R_{S_1} \right] \left[ \overline{^{B_2}S_2, ^{B2}R_{S_2}} \right] \right] \quad \left[ z_m \quad R_m \right]
                                         \mathcal{H}_p = \begin{bmatrix} 0 & 1 \end{bmatrix}
                  B_2 x_{B_0} = (\ominus^N x_{B_2}) \oplus {}^N x_{B_0};
\boxed{[\boldsymbol{z_{p_1}}, \boldsymbol{R_{p_1}}] = Register(^{\boldsymbol{B_2}}\boldsymbol{S_2}, ^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_2}}\boldsymbol{x_{B_0}})}
```

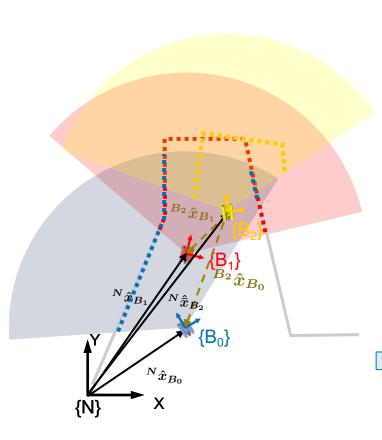


```
[^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                            [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                                                                                                                                                                                                                                                                     // Grow the state vector
                          \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                                                                                                                                                                                                                                                                // Store the scan in the map
                          for k = 1 to steps do
                                                [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                                                                                                                                                                                                                                            // Get input to the motion model
                                               [\hat{N}\hat{\bar{x}}_{k}, \hat{N}\bar{P}_{k}] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_{k}, Q_{k});
                                               [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                                                                                                                                                                                                                                                           // Read navigation sensors
                                              oldsymbol{z_{p}} = [\ ]; oldsymbol{z_{p}} = [\ ]; \mathcal{H}_{p} = [\ ];
                                              if ScanAvailable then
                                                                   [^{\boldsymbol{B_k}}\boldsymbol{S_k}, ^{\boldsymbol{B_k}}\boldsymbol{R_{S_k}}] = GetScan();
                                                                 [^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k}] = AddNewPose(^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k});
                                                                                                                                                                                                                                                                                                                                                                                                                     // Grow the state vector
                                                                \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                                                                                                                                                                                                                                                                // Store the scan in the map
                                                               \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                                                                                                                                                                                                                                                                                       // Get pairs of overlapping scans
                                                               c = 1; \mathcal{H}_p = [\ ];
                                                                 for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                                                                                                                                                                                             ^{\prime\prime} for all overlaps ^{\prime\prime} // Get the pair: ^{\prime\prime} ^{\prime\prime} overlaps ^{\prime\prime} ^
                                                                                   j = \mathcal{H}_o[i];
                                                                                \begin{bmatrix} B_{j} S_{j}, B_{j} R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; 
 \begin{bmatrix} B_{j} X_{j}, B_{j} R_{S_{j}} \end{bmatrix} = \mathcal{M}[j]; 
 B_{j} X_{B_{k}} = (\ominus^{N} X_{B_{j}}) \oplus^{N} X_{B_{k}}; 
 B_{j} P_{B_{k}} = J_{1} \oplus J_{\ominus}^{N} P_{B_{j}} J_{\ominus}^{T} J_{1}^{T} \oplus + J_{2} \oplus^{N} P_{B_{k}} J_{2}^{T} \oplus 
                                                                                                                                                                                                                                                                                                                                                                                         // Get the scans from the map
                                                                                                                                                                                                                                                                                                                                                                                                          // Scan Displacement guess
                                                                                       [\boldsymbol{z_r}, \boldsymbol{R_r}] = Register(\check{B_j} \boldsymbol{S_i}, \check{B_k} \boldsymbol{S_k}, \check{B_j} \boldsymbol{x_{B_i}}):
                                                                                                                                                                                                                                                                                                                                                                      // Scan displacement mean & cov
                                                                                  if IndividuallyCompatible(B_j x_{B_l}, B_j P_{B_l}, z_r, R_r, \alpha) then
                                                                                                     \boldsymbol{z_p}[c] = \boldsymbol{z_r}; \ \boldsymbol{R_p}[c] = \boldsymbol{R_r};
                                                                                                                                                                                                                                                                                                                                                                                                                     // Accepted registration
                                                                                          \mathcal{H}_{n}[c] = i; c = c + 1;
                                            [\mathbf{z_k}, \mathbf{R_k}, \mathbf{H_k}, \mathbf{V_k}] = ObservationMatrix(\mathcal{H}_p, {}^{\mathbf{N}}\mathbf{\hat{\bar{z}}_k}, \mathbf{z_m}, \mathbf{R_m}, \mathbf{z_p}, \mathbf{R_p}); \\ [{}^{\mathbf{N}}\mathbf{\hat{x}_k}, {}^{\mathbf{N}}\mathbf{P_k}] = Update({}^{\mathbf{N}}\mathbf{\hat{z}_k}, {}^{\mathbf{N}}\mathbf{\bar{P}_k}, \mathbf{z_k}, \mathbf{R_k}, \mathbf{H_k}, \mathbf{V_k}, \mathcal{H}_p);
                          {}^{N}\hat{\bar{x}}_{2} = \begin{bmatrix} {}^{N}\hat{x}_{B_{0}} \\ {}^{N}\hat{x}_{B_{1}} \\ {}^{N}\hat{\bar{x}}_{B_{2}} \\ {}^{N}\hat{\bar{x}}_{B_{2}} \end{bmatrix} \; ; \; {}^{N}\bar{P}_{2} = \begin{bmatrix} {}^{N}P_{B_{0}} & {}^{N}P_{B_{0}B_{1}} & {}^{N}\bar{P}_{B_{0}B_{2}} & {}^{N}\bar{P}_{B_{0}B_{2}} \\ {}^{N}P_{B_{1}B_{0}} & {}^{N}P_{B_{1}} & {}^{N}\bar{P}_{B_{1}B_{2}} & {}^{N}\bar{P}_{B_{1}B_{2}} \\ {}^{N}\bar{P}_{B_{2}B_{0}} & {}^{N}\bar{P}_{B_{2}B_{1}} & {}^{N}\bar{P}_{B_{2}} & {}^{N}\bar{P}_{B_{2}} \\ {}^{N}\bar{P}_{B_{2}B_{0}} & {}^{N}\bar{P}_{B_{2}B_{1}} & {}^{N}\bar{P}_{B_{2}} & {}^{N}\bar{P}_{B_{2}} \end{bmatrix}
                                  \mathcal{M}_2 = \left[ \begin{bmatrix} B_0 S_0, B^0 R_{S_0} \end{bmatrix} \begin{bmatrix} \overline{B_1} S_1, \overline{B_1} R_{S_1} \end{bmatrix} \begin{bmatrix} \overline{B_2} S_2, \overline{B_2} R_{S_2} \end{bmatrix} \right] \quad \begin{bmatrix} \boldsymbol{z_m} & R_m \end{bmatrix}
                                    \mathcal{H}_{p} = \begin{bmatrix} 0 & 1 \end{bmatrix}
                [z_{p_1}, R_{p_1}] = Register(^{B_2}S_2, ^{B_0}S_0, ^{\overline{B_2}}x_{B_0}) ; [z_{p_2}, R_{p_2}] = Register(^{B_2}S_2, ^{B_1}S_1, ^{B_2}x_{B_1})
                                          oldsymbol{z_p} = egin{bmatrix} oldsymbol{z_{p_1}} & oldsymbol{z_{p_2}} \end{bmatrix} \; ; \; oldsymbol{R_p} = egin{bmatrix} oldsymbol{R_{p_1}} & oldsymbol{R_{p_2}} \end{bmatrix}
```

// Initialize SLAM state vector

// Read the Scan from the sensor

Method Localization  $(\hat{x}_0, P_0)$  $[{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];$ 



```
egin{aligned} egin{aligned} oldsymbol{z_k} &= egin{bmatrix} oldsymbol{z_m} \ oldsymbol{z_p} \end{bmatrix} \; ; \; oldsymbol{R_k} &= egin{bmatrix} oldsymbol{H_m} & 0 \ 0 & oldsymbol{H_p} \end{bmatrix} \; ; \; oldsymbol{V_k} &= egin{bmatrix} oldsymbol{I} & 0 \ 0 & oldsymbol{I} \end{bmatrix} \end{aligned}
```

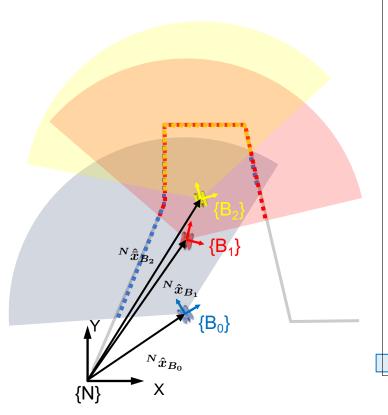
```
Method Localization (\hat{x}_0, P_0)
        [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                     // Initialize SLAM state vector
        [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                    // Read the Scan from the sensor
       [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                    // Grow the state vector
       \mathcal{M} = \left[ \left[ B_0 S_0, B_0 R_{S_0} \right] \right];
                                                                                                                                                           // Store the scan in the map
       for k = 1 to steps do
                [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                   // Get input to the motion model
                [N\hat{\bar{x}}_k, N\bar{P}_k] = Prediction(N\hat{x}_{k-1}, NP_{k-1}, u_k, Q_k);
               [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                                // Read navigation sensors
               oldsymbol{z_p} = [\ ]; oldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
               if ScanAvailable then
                       [^{\boldsymbol{B_k}}\boldsymbol{S_k}, ^{\boldsymbol{B_k}}\boldsymbol{R_{S_k}}] = GetScan();
                       [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = AddNewPose({}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k});
                                                                                                                                                                    // Grow the state vector
                      \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                           // Store the scan in the map
                      \mathcal{H}_o = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M}):
                                                                                                                                                 // Get pairs of overlapping scans
                      c = 1; \mathcal{H}_p = [\ ];
                      for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                              // for all overlaps
                              j = \mathcal{H}_o[i];
                                                                                                                                       // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                              \begin{split} &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}x_{B_k} = (\ominus^Nx_{B_j}) \oplus^Nx_{B_k}; \\ &[^{B_j}P_{B_k} = J_{1\oplus}J_{\ominus}^NP_{B_j}J_{\ominus}^TJ_{1\oplus}^T + J_{2\oplus}^NP_{B_k}J_{2\oplus}^T \end{split} 
                                                                                                                                                         // Get the scans from the map
                                                                                                                                                                // Scan Displacement guess
                                [\boldsymbol{z_r}, \boldsymbol{R_r}] = Register(\overset{\boldsymbol{B_j}}{S_i}, \overset{\boldsymbol{T_k}}{S_k}, \overset{\boldsymbol{B_j}}{S_k}, \overset{\boldsymbol{B_j}}{S_k}, \overset{\boldsymbol{B_j}}{S_k}):
                                                                                                                                                    // Scan displacement mean & cov
                              if IndividuallyCompatible(B_j x_{B_k}, B_j P_{B_k}, z_r, R_r, \alpha) then
                                     \boldsymbol{z_n}[c] = \boldsymbol{z_r}; \ \boldsymbol{R_n}[c] = \boldsymbol{R_r};
                                                                                                                                                                    // Accepted registration
                                 \mathcal{H}_{n}[c] = i; c = c + 1;

\overline{[\boldsymbol{z}_{k}, \boldsymbol{R}_{k}, \boldsymbol{H}_{k}, \boldsymbol{V}_{k}]} = ObservationMatrix(\mathcal{H}_{p}, {}^{N}\boldsymbol{\hat{x}}_{k}, \boldsymbol{z}_{m}, \boldsymbol{R}_{m}, \boldsymbol{z}_{p}, \boldsymbol{R}_{p});

\overline{[}^{N}\boldsymbol{\hat{x}}_{k}, {}^{N}\boldsymbol{P}_{k}]} = Update({}^{N}\boldsymbol{\hat{x}}_{k}, {}^{N}\boldsymbol{\bar{P}}_{k}, \boldsymbol{z}_{k}, \boldsymbol{R}_{k}, \boldsymbol{H}_{k}, \boldsymbol{V}_{k}, \mathcal{H}_{p});
```

 $N_{\hat{x}_{2}} = \begin{bmatrix} N_{\hat{x}_{B_{0}}} \\ N_{\hat{x}_{B_{1}}} \\ N_{\hat{x}_{B_{2}}} \\ N_{\hat{x}_{B_{2}}} \end{bmatrix}; N_{\bar{P}_{2}} = \begin{bmatrix} N_{P_{B_{0}}} & N_{P_{B_{0}B_{1}}} & N_{\bar{P}_{B_{0}B_{2}}} & N_{\bar{P}_{B_{0}B_{2}}} \\ N_{P_{B_{1}B_{0}}} & N_{P_{B_{1}}} & N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{1}B_{2}}} \\ N_{\bar{P}_{B_{2}B_{0}}} & N_{\bar{P}_{B_{2}B_{1}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{2}B_{1}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{2}B_{1}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{1}B_{2}}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}} \\ N_{\bar{P}_{B_{2}B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{2}B_{2}} & N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}}} \\ N_{\bar{P}_{B_{2}}} & N_{\bar{P}_{B_{2}} & N_{\bar{P}_{B_{2}}$ 

 $oldsymbol{z_{n}} = egin{bmatrix} oldsymbol{z_{n}} & oldsymbol{z_{n}} \end{bmatrix} \; ; \; oldsymbol{R_{n}} = egin{bmatrix} oldsymbol{R_{n_1}} & oldsymbol{R_{n_2}} \end{bmatrix}$ 



```
Method Localization (\hat{x}_0, P_0)
        [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                                                                  // Initialize SLAM state vector
       [^{\boldsymbol{B_0}}\boldsymbol{S_0}, ^{\boldsymbol{B_0}}\boldsymbol{R_{S_0}}] = GetScan();
                                                                                                                                                // Read the Scan from the sensor
       [{}^{N}\hat{\bar{x}}_{0}, {}^{N}P_{0}] = AddNewPose({}^{N}\hat{x}_{0}, {}^{N}\bar{P}_{0});
                                                                                                                                                                 // Grow the state vector
      \mathcal{M} = [[^{B_0}S_0, ^{B_0}R_{S_0}]];
                                                                                                                                                        // Store the scan in the map
      for k = 1 to steps do
               [\boldsymbol{u_k}, \boldsymbol{Q_k}] = GetInput();
                                                                                                                                                // Get input to the motion model
               [\hat{N}\hat{\bar{x}}_k, \hat{N}\bar{P}_k] = Prediction(\hat{N}\hat{x}_{k-1}, \hat{N}P_{k-1}, u_k, Q_k);
               [\boldsymbol{z_m}, \boldsymbol{R_m}] = GetMeasurement();
                                                                                                                                                            // Read navigation sensors
              \boldsymbol{z_p} = [\ ]; \boldsymbol{z_p} = [\ ]; \mathcal{H}_p = [\ ];
              if ScanAvailable then
                      [^{B_k}S_k, ^{B_k}R_{S_k}] = GetScan();
                      [{}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k}] = AddNewPose({}^{N}\hat{\bar{x}}_{k}, {}^{N}\bar{P}_{k});
                                                                                                                                                                // Grow the state vector
                      \mathcal{M}[k] = [^{B_k} S_k, ^{B_k} R_{S_k}];
                                                                                                                                                        // Store the scan in the map
                     \mathcal{H}_{o} = OverlappingScans(^{N}\hat{\bar{x}}_{k}, \mathcal{M});
                                                                                                                                              // Get pairs of overlapping scans
                     c = 1; \mathcal{H}_p = [\ ];
                      for i = 1 to length(\mathcal{H}_o) do
                                                                                                                                                                          // for all overlaps
                             j = \mathcal{H}_o[i];
                                                                                                                                   // Get the pair: {}^{B_j}S_i overlaps {}^{B_k}S_k
                             \begin{aligned} &[^{B_j}S_j,^{B_j}R_{S_j}] = \mathcal{M}[j]; \\ &[^{B_j}x_{B_k} = (\ominus^Nx_{B_j}) \oplus^Nx_{B_k}; \\ &[^{B_j}P_{B_k} = J_1 \oplus J_{\ominus}^N P_{B_j}J_{\ominus}^T J_{1\oplus}^T + J_{2\oplus}^N P_{B_k}J_{2\oplus}^T \\ &[z_r, R_r] = Register(^{B_j}S_j,^{B_k}S_k,^{B_j}x_{B_k}); \end{aligned} 
                                                                                                                                                     // Get the scans from the map
                                                                                                                                                            // Scan Displacement guess
                                                                                                                                                // Scan displacement mean & cov
                             if IndividuallyCompatible(B_j x_{B_k}, B_j P_{B_k}, z_r, R_r, \alpha) then
                                    oldsymbol{z_p}[c] = oldsymbol{z_r}; \ oldsymbol{R_p}[c] = oldsymbol{R_r};
                                                                                                                                                                // Accepted registration
                                \mathcal{H}_n[c] = i; c = c + 1;
                \begin{aligned}        [\boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}] &= Observation Matrix(\mathcal{H}_p, {}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, \boldsymbol{z_m}, \boldsymbol{R_m}, \boldsymbol{z_p}, \boldsymbol{R_p}); \\       [{}^{\boldsymbol{N}}\boldsymbol{\hat{x}_k}, {}^{\boldsymbol{N}}\boldsymbol{P_k}] &= Update({}^{\boldsymbol{N}}\boldsymbol{\hat{\bar{x}}_k}, {}^{\boldsymbol{N}}\boldsymbol{\bar{P}_k}, \boldsymbol{z_k}, \boldsymbol{R_k}, \boldsymbol{H_k}, \boldsymbol{V_k}, \mathcal{H}_p); \end{aligned}
```

$$egin{aligned} N\hat{x}_2 &= egin{bmatrix} N\hat{x}_{B_0} \ N\hat{x}_{B_1} \ N\hat{x}_{B_2} \ N\hat{x}_{B_2} \end{bmatrix}; \ NP_2 &= egin{bmatrix} NP_{B_0} & NP_{B_0B_1} & NP_{B_0B_2} \ NP_{B_1B_0} & NP_{B_1} & NP_{B_1,B_2} \ NP_{B_2B_0} & NP_{B_2B_1} & NP_{B_2} \ NP_{B_2B_0} & NP_{B_2B_1} & NP_{B_2} \ NP_{B_2} \end{bmatrix} \ \mathcal{M}_2 &= egin{bmatrix} [B_0S_0, B_0R_{S_0}] \ [B_1S_1, B_1R_{S_1}] \ [B_2S_2, B_2R_{S_2}] \end{bmatrix} \end{aligned}$$

 $^{N}\hat{x}_{B_{0}}$ 

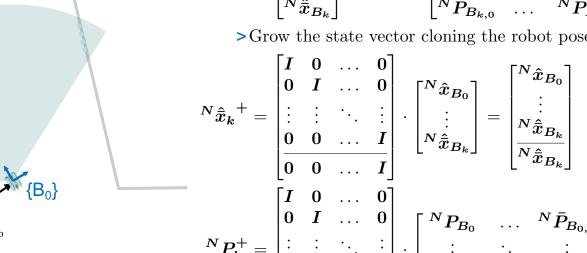


### Add a New Viewpoint Pose

> Previous State Vector

$$egin{aligned} {}^{N}\hat{ar{x}}_{k} = egin{bmatrix} {}^{N}\hat{ar{x}}_{B_0} \ \vdots \ {}^{N}ar{P}_{k} = egin{bmatrix} {}^{N}P_{B_0} & \dots & {}^{N}ar{P}_{B_{0,k}} \ \vdots & \ddots & \vdots \ {}^{N}ar{P}_{B_{k,0}} & \dots & {}^{N}ar{P}_{B_k} \end{bmatrix} \end{aligned}$$

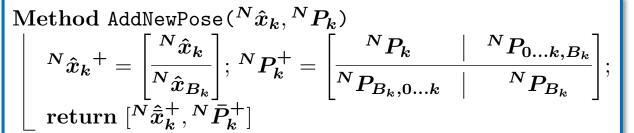
> Grow the state vector cloning the robot pose



 $\{B_0\}$ 



## Add a New Viewpoint Pose



$$egin{aligned} {}^{N}\hat{ar{x}}_{k}^{+} &= egin{bmatrix} I & 0 & \dots & 0 \ 0 & I & \dots & 0 \ dots & dots & \ddots & dots \ \dfrac{1}{0} & 0 & \dots & I \ 0 & 0 & \dots & I \end{bmatrix} \cdot egin{bmatrix} N\hat{x}_{B_{0}} \ dots \ N\hat{x}_{B_{k}} \end{bmatrix} = egin{bmatrix} N\hat{x}_{B_{0}} \ dots \ N\hat{x}_{B_{k}} \end{bmatrix} \ NP_{k}^{+} &= egin{bmatrix} I & 0 & \dots & 0 \ 0 & I & \dots & 0 \ 0 & I & \dots & 0 \ dots & dots & \ddots & dots \ Nar{P}_{B_{0}} & \dots & Nar{P}_{B_{0},k} \ dots & \ddots & \ddots & dots \ 0 & 0 & \dots & I \ \end{bmatrix} \cdot egin{bmatrix} I & 0 & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & 0 \ 0 & I & \dots & 0 & I \ \end{bmatrix} \ NP_{B_{0}} & \dots & Nar{P}_{B_{0,k}} \end{bmatrix}$$



### **Prediction**

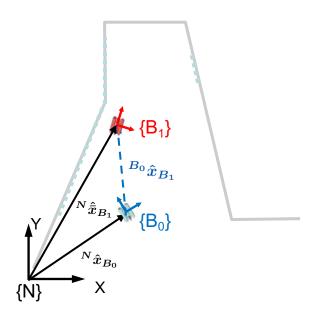


$${}^{N}\hat{x}_{k-1} = \begin{bmatrix} {}^{N}\hat{x}_{B_0} \\ \vdots \\ {}^{N}\hat{x}_{B_{k-2}} \\ \hline {}^{N}\hat{x}_{B_{k-1}} \end{bmatrix}; {}^{N}P_{k-1} = \begin{bmatrix} {}^{N}P_{B_0} & \dots & {}^{N}P_{B_{0,k-1}} \\ \vdots & \ddots & \vdots \\ {}^{N}P_{B_{k-1,0}} & \dots & {}^{N}P_{B_{k-1}} \end{bmatrix}$$

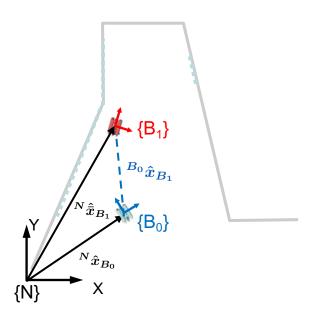
$$\geq \operatorname{Predicted State Vector}$$

> Predicted State Vector

$$^{N}ar{P}_{k} = F_{1_{k}}{}^{N}P_{k-1}F_{1_{k}}^{T} + F_{2_{k}}\overline{Q_{k}F_{2_{k}}^{T}}$$







### **Prediction**

> Previous State Vector

$${}^{N}\hat{x}_{k-1} = egin{bmatrix} {}^{N}\hat{x}_{B_0} \ \vdots \ {}^{N}\hat{x}_{B_{k-2}} \ {}^{N}\hat{x}_{B_{k-1}} \end{bmatrix} \; ; \; {}^{N}P_{k-1} = egin{bmatrix} {}^{N}P_{B_0} & \dots & {}^{N}P_{B_{0,k-1}} \ \vdots & \ddots & \vdots \ {}^{N}P_{B_{k-1,0}} & \dots & {}^{N}P_{B_{k-1}} \end{bmatrix}$$

> Predicted State Vector

$$egin{aligned} ^{N}\hat{ar{x}}_{k} &= \mathfrak{f}(^{N}\hat{x}_{k-1},u_{k},\hat{w}_{k}) = egin{bmatrix} ^{N}\hat{x}_{B_{0}} \ &dots \ ^{N}\hat{x}_{B_{k-2}} \ f(^{N}\hat{x}_{B_{k-1}},u_{k},\hat{w}_{k}) \end{bmatrix} \end{aligned}$$

$$F_{1_k} = \frac{\partial \mathfrak{f}(^N x_{k-1}, u_k, w_k)}{\partial^N x_{k-1}}$$

$$= \begin{bmatrix} \frac{\partial^N x_{B_0}}{\partial^N x_{B_0}} & \frac{\partial^N x_{B_0}}{\partial^N x_{B_1}} & \cdots & \frac{\partial^N x_{B_0}}{\partial^N x_{B_{k-2}}} & \frac{\partial^N x_{B_0}}{\partial^N x_{B_{k-1}}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_0}} & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_1}} & \cdots & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_{k-2}}} & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_{k-1}}} \\ \frac{\partial f(^N x_{B_{k-1}}, u_k, w_k)}{\partial^N x_{B_0}} & \frac{\partial f(^N x_{B_{k-1}}, u_k, w_k)}{\partial^N x_{B_1}} & \cdots & \frac{\partial f(^N x_{B_{k-1}}^T, u_k^T, w_k)}{\partial^N x_{B_{k-2}}} & \frac{\partial f(^N x_{B_{k-1}}^T, u_k^T, w_k)}{\partial^N x_{B_{k-1}}} \end{bmatrix}$$

$$= \begin{bmatrix} I & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & I & 0 \\ 0 & 0 & 0 & I_t \end{bmatrix}$$



### **Prediction**

> Previous State Vector

$${}^{N}\hat{x}_{k-1} = egin{bmatrix} {}^{N}\hat{x}_{B_0} \ dots \ {}^{N}\hat{x}_{B_{k-2}} \ {}^{N}\hat{x}_{B_{k-1}} \end{bmatrix} \; ; \; {}^{N}P_{k-1} = egin{bmatrix} {}^{N}P_{B_0} & \dots & {}^{N}P_{B_{0,k-1}} \ dots & \ddots & dots \ {}^{N}P_{B_{k-1,0}} & \dots & {}^{N}P_{B_{k-1}} \end{bmatrix}$$

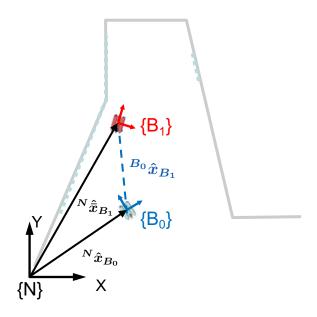
> Predicted State Vector

$$egin{align*} N \hat{ar{x}}_k &= \mathfrak{f}(^N \hat{x}_{k-1}, u_k, \hat{w}_k) = egin{bmatrix} N \hat{x}_{B_0} & dots & dots \ N \hat{x}_{B_{k-2}} & dots \ f(^N \hat{x}_{B_{k-1}}, u_k, \hat{w}_k) \end{bmatrix} \ N ar{P}_k &= F_{1_k}{}^N P_{k-1} F_{1_k}^T + F_{2_k} Q_k F_{2_k}^T \ & dots & dots \ rac{\partial^N x_{B_0}}{\partial w_k} & dots \ rac{dots}{\partial w_k} & dots \ rac{\partial^N x_{B_{k-2}}}{\partial w_k} & dots \ rac{\partial^N x_{B_{k-2}}}{\partial w_k} & dots \ rac{\partial^N x_{B_{k-2}}}{\partial w_k} & dots \ rac{\partial^N x_{B_{k-1}}}{\partial w_k} & dots \ rac{\partial^N x_{B_{k-1}}}{\partial w_k} & dots \ \end{pmatrix} = egin{bmatrix} 0 & dots & dots \ 0 &$$

$$F_{1_k} = \frac{\partial \mathfrak{f}(^N x_{k-1}, u_k, w_k)}{\partial^N x_{k-1}}$$

$$= \begin{bmatrix} \frac{\partial^N x_{B_0}}{\partial^N x_{B_0}} & \frac{\partial^N x_{B_0}}{\partial^N x_{B_1}} & \cdots & \frac{\partial^N x_{B_0}}{\partial^N x_{B_{k-2}}} & \frac{\partial^N x_{B_0}}{\partial^N x_{B_{k-1}}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_0}} & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_1}} & \cdots & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_{k-2}}} & \frac{\partial^N x_{B_{k-2}}}{\partial^N x_{B_{k-1}}} \\ \frac{\partial f(^N x_{B_{k-1}}, u_k, w_k)}{\partial^N x_{B_0}} & \frac{\partial f(^N x_{B_{k-1}}, u_k, w_k)}{\partial^N x_{B_1}} & \cdots & \frac{\partial f(^N x_{B_{k-1}}^T, u_k^T, w_k)}{\partial^N x_{B_{k-2}}} & \frac{\partial f(^N x_{B_{k-1}}^T, u_k^T, w_k)}{\partial^N x_{B_{k-1}}} \end{bmatrix}$$

$$= \begin{bmatrix} I & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & I & 0 \\ 0 & \dots & 0 & I_t \end{bmatrix}$$





### **Prediction**

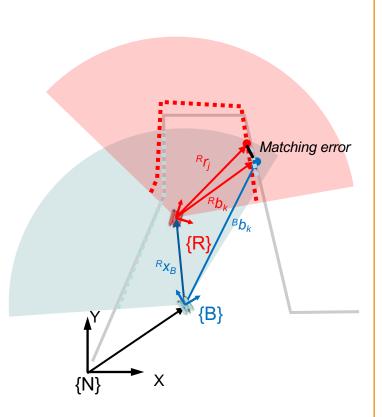
$$\begin{split} {}^{N}\bar{P}_{k} &= F_{1_{k}}{}^{N}P_{k-1}F_{1_{k}}^{T} + F_{2_{k}}Q_{k}F_{2_{k}}^{T} \\ &= \begin{bmatrix} I & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & I & 0 \\ 0 & \dots & 0 & J_{f_{w}} \end{bmatrix} \begin{bmatrix} {}^{N}P_{B_{0}} & \dots & {}^{N}P_{B_{0,k-1}} \\ \vdots & \ddots & \vdots & \vdots \\ {}^{N}P_{B_{k-1,0}} & \dots & {}^{N}P_{B_{k-1}} \end{bmatrix} \begin{bmatrix} I & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & I & 0 \\ 0 & \dots & 0 & J_{f_{w}}^{T} \end{bmatrix} \\ &+ \begin{bmatrix} 0 \\ \vdots \\ 0 \\ J_{f_{w}} \end{bmatrix} \begin{bmatrix} Q_{k} \begin{bmatrix} 0 & 0 & \dots & J_{f_{w}}^{T} \end{bmatrix} \\ Q_{k} \begin{bmatrix} 0 & 0 & \dots & J_{f_{w}}^{T} \end{bmatrix} \\ &\vdots & \ddots & \vdots & \vdots \\ N^{N}P_{B_{k-1,0}} & \dots & N^{N}P_{B_{k-2}} & N^{N}P_{B_{k-1,k-2}} J_{f_{w}}^{T} \\ J_{f_{x}}{}^{N}P_{B_{k-1,0}} & \dots & J_{f_{x}}{}^{N}P_{B_{k-1,k-2}} \end{bmatrix} J_{f_{x}}{}^{N}P_{B_{k-1}}J_{f_{x}}^{T} \\ &= \begin{bmatrix} N^{N}P_{B_{0...k-2,0...k-2}} & N^{N}P_{B_{0...k-2,k-1}}J_{f_{w}}^{T} \\ J_{f_{x}}{}^{N}P_{B_{k-1,0...k-2}} & J_{f_{x}}{}^{N}P_{B_{k-1,1}}J_{f_{x}}^{T} + J_{f_{w}}Q_{k}J_{f_{w}}^{T} \end{bmatrix}. \end{split}$$



### **Prediction**

$$= egin{bmatrix} ^N P_{B_{0...k-2,0...k-2}} & ^N P_{B_{0...k-2,k-1}} J_{f_x}^T \ \hline J_{f_x}^N P_{B_{k-1,0...k-2}} & ^I J_{f_x}^N P_{B_{k-1}} J_{f_x}^T + J_{f_w} Q_k J_{f_w}^T \end{bmatrix}.$$





```
Function ICP({}^{\mathcal{R}}S_r, {}^{\mathcal{B}}S_b, {}^{\mathcal{R}}x_B))

// New Scan: {}^{\mathcal{R}}S_r = [{}^{\mathcal{R}}r_1 {}^{\mathcal{R}}r_2 \dots {}^{\mathcal{R}}r_N]

// Reference Scan: {}^{\mathcal{B}}S_b = [{}^{\mathcal{B}}b_1 {}^{\mathcal{B}}b_2 \dots {}^{\mathcal{B}}b_S]

// Initial Transformation guess: {}^{\mathcal{R}}x_B

i=1;

while (i < maxIterations) and (not\ converged) do

for j=1 to N do

for j=1 to N do

{}^{\mathcal{B}}c_j = \underset{{}^{\mathcal{B}}b_k \in {}^{\mathcal{B}}S_b}{\operatorname{sgmin}}\{||{}^{\mathcal{R}}x_B \oplus {}^{\mathcal{B}}b_k - {}^{\mathcal{R}}r_j||\};

end

// Compute the Nearest Neighbor

{}^{\mathcal{R}}x_B = \underset{{}^{\mathcal{R}}x_B}{\operatorname{argmin}}\{\sum_j^N e_j^T \cdot e_j\} where e_j = {}^{\mathcal{R}}x_B \oplus {}^{\mathcal{B}}c_j - {}^{\mathcal{R}}r_j;

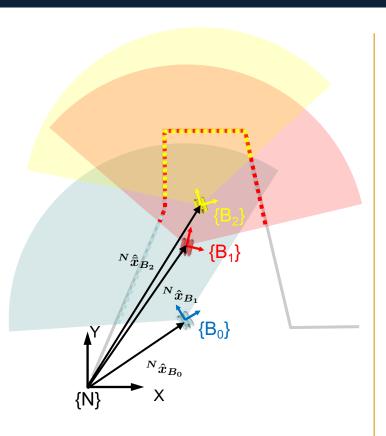
i=i+1;

end

return {}^{\mathcal{R}}x_B;
```

Algorithm 1: Iterative Closest Point





 $B_0$ -frame  $\equiv \mathcal{B}$ -frame  $B_1$ -frame  $\equiv \mathcal{R}$ -frame  $B_2$ -frame  $\equiv \mathcal{Y}$ -frame

### **Pose Constraints**

1. Compute overlapping scans

$$\mathcal{H}_p = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M})$$
$$\mathcal{H}_p = [0 \ 1 \ 2]$$

2. Register overlapping scans

$$\begin{bmatrix}
[B_{k} x_{y}, B_{k} R_{y}] \\
[B_{k} x_{y}, B_{k} R_{y}]
\end{bmatrix} = Registration(B_{k} S_{k}, S_{y})$$

$$\begin{bmatrix}
[B_{k} x_{y}, B_{k} R_{y}] \\
[B_{k} x_{y}, B_{k} R_{y}]
\end{bmatrix} = Registration(B_{k} S_{k}, S_{y})$$

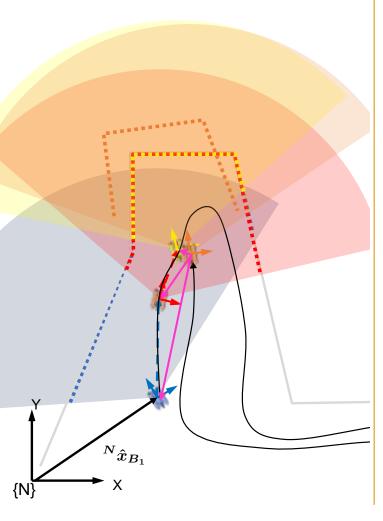
$$\begin{bmatrix}
[B_{k} x_{y}, B_{k} R_{y}] \\
[B_{k} x_{y}, B_{k} R_{y}]
\end{bmatrix} = Registration(B_{k} S_{k}, S_{y})$$

3. Observation equation

$$egin{aligned} egin{aligned} egi$$



## **Loop Closing**



### **Pose Constraints**

1. Compute overlapping scans

$$\mathcal{H}_p = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M})$$
$$\mathcal{H}_p = [0 \ 1 \ 2]$$

2. Register overlapping scans

$$\begin{bmatrix} B_{k} x_{\mathcal{Y}}, B_{k} R_{\mathcal{Y}} \end{bmatrix} = Registration(B_{k} S_{k}, \mathcal{Y} S_{y})$$
$$\begin{bmatrix} B_{k} x_{\mathcal{B}}, B_{k} R_{\mathcal{B}} \end{bmatrix} = Registration(B_{k} S_{k}, \mathcal{B} S_{b})$$
$$\begin{bmatrix} B_{k} x_{\mathcal{R}}, B_{k} R_{\mathcal{R}} \end{bmatrix} = Registration(B_{k} S_{k}, \mathcal{B} S_{r}),$$

3. Observation equation

$$egin{aligned} egin{aligned} egi$$



### **Observation Jacobians**

$$H_{k} = \frac{\partial h(^{N}\bar{x}_{k}, v_{k})}{\partial^{N}\bar{x}_{k}}\bigg|_{N_{\bar{x}_{k}} = N_{\bar{x}_{k}}^{2}, w_{k} = 0}$$

$$= \begin{bmatrix}
1 & 2 & y & b & r & n_{p} \\
1 & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_{y_{k}})}{\partial^{N}x_{B_{0}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_{y_{k}})}{\partial^{N}x_{B_{1}}} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_{y_{k}})}{\partial^{N}xy} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_{y_{k}})}{\partial^{N}x_{B_{1}}} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_{y_{k}})}{\partial^{N}xy} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}})\ominus^{N}xy + v_$$

$$\begin{split} V_{k} &= \left. \frac{\partial h(^{N}\bar{x}_{k}, v_{k})}{\partial v_{k}} \right|_{\bar{x}_{k} = ^{N}\hat{\bar{x}}_{k}, w_{k} = 0} \\ &= \begin{bmatrix} \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial v_{y_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial v_{b_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial v_{r_{k}}} \\ \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{b_{k}})}{\partial v_{y_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{b_{k}})}{\partial v_{b_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{b_{k}})}{\partial v_{r_{k}}} \\ \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{r_{k}})}{\partial v_{y_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{r_{k}})}{\partial v_{b_{k}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{B}} + v_{r_{k}})}{\partial v_{r_{k}}} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \end{split}$$



### **Observation Jacobians**

$$H_{k} = \frac{\partial h(^{N}\bar{x}_{k}, v_{k})}{\partial^{N}\bar{x}_{k}} \bigg|_{N_{\bar{x}_{k}} = N_{\bar{x}_{k}}, w_{k} = 0}$$

$$= \begin{bmatrix} 1 & 2 & \cdots & y \\ 1 & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{B_{0}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{B_{1}}} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{y_{0}}} \\ 3 & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{R}} + v_{v_{k}})}{\partial^{N} x_{B_{0}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{B_{1}}} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{\mathcal{Y}}} \\ 3 & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{R}} + v_{r_{k}})}{\partial^{N} x_{B_{0}}} & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{R}} + v_{r_{k}})}{\partial^{N} x_{B_{1}}} & \cdots & \frac{\partial((\ominus^{N}\bar{x}_{B_{k}}) \oplus^{N} x_{\mathcal{Y}} + v_{y_{k}})}{\partial^{N} x_{\mathcal{Y}}} \\ = \begin{bmatrix} 1 & 2 & \cdots & y & \cdots & b & \cdots & r & \cdots & n_{p} \\ 1 & 0 & 0 & \cdots & J_{2 \oplus} & \cdots & \cdots & \cdots & J_{1 \oplus} J_{\ominus} \\ 2 & 0 & 0 & \cdots & \cdots & \cdots & J_{2 \oplus} & \cdots & \cdots & J_{1 \oplus} J_{\ominus} \\ 3 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & J_{2 \oplus} & \cdots & J_{1 \oplus} J_{\ominus} \end{bmatrix}$$

$$\begin{array}{l} \text{Method ObservationMatrix}(\mathcal{H}_p, {}^N\hat{\bar{x}}_k, z_m, R_m, z_p, R_p) \\ & H_p = 0; \\ & \text{for } i = 1 \ to \ length(z_p) \ \text{do} \\ & \left\lfloor \begin{array}{l} H_p[i,1] = J_{1\oplus}(\ominus^N\hat{\bar{x}}_{B_k}, {}^N\hat{x}_k[\mathcal{H}_p[i]])J_{\ominus}({}^N\hat{\bar{x}}_{B_k}); \\ H_p[i,\mathcal{H}_p[i]] = J_{2\oplus}({}^N\hat{\bar{x}}_{B_k}); \\ \end{array} \right. \\ & \left\lfloor \begin{array}{l} z_m \\ z_p \end{array} \right\rfloor; R_k = \left[ \begin{array}{l} R_m & 0 \\ 0 & R_p \end{array} \right]; \\ & H_k = \left[ \begin{array}{l} J_{h_x}(\hat{\bar{x}}_k) \\ H_p \end{array} \right]; \\ & V_k = \left[ \begin{array}{l} J_{h_v}(\hat{\bar{x}}_k) \\ V_p \end{array} \right]; \\ & \text{return } [z_k, R_k, H_k, V_k]; \end{array}$$

$$\begin{split} & \left[ \frac{\partial h(^N \bar{x}_k, v_k)}{\partial v_k} \right|_{\bar{x}_k = ^N \hat{\bar{x}}_k, w_k = 0} \\ & = \begin{bmatrix} \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{Y}} + v_{y_k} \right)}{\partial v_{y_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{Y}} + v_{y_k} \right)}{\partial v_{b_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{Y}} + v_{y_k} \right)}{\partial v_{b_k}} \\ \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{B}} + v_{b_k} \right)}{\partial v_{y_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{B}} + v_{b_k} \right)}{\partial v_{b_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{B}} + v_{b_k} \right)}{\partial v_{b_k}} \\ \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{R}} + v_{r_k} \right)}{\partial v_{b_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{B}} + v_{b_k} \right)}{\partial v_{b_k}} & \frac{\partial \left( (\ominus^N \bar{x}_{B_k}) \oplus^N x_{\mathcal{B}} + v_{b_k} \right)}{\partial v_{r_k}} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \end{aligned}$$



## **Update**

1. The Standard EKF Update equations are used:

$$egin{aligned} oldsymbol{K}_k &= ^{oldsymbol{N}} ar{oldsymbol{P}}_k oldsymbol{H}_k^T (oldsymbol{H}_k^{oldsymbol{N}} ar{oldsymbol{P}}_k oldsymbol{H}_k^T + oldsymbol{V}_k oldsymbol{R}_k oldsymbol{V}_k^T)^{-1} \ ^{oldsymbol{N}} \hat{oldsymbol{x}}_k &= ^{oldsymbol{N}} \hat{oldsymbol{x}}_k + oldsymbol{K}_k \left( oldsymbol{z}_k - oldsymbol{h} (^{oldsymbol{N}} \hat{oldsymbol{x}}_k, oldsymbol{z}_p, \mathcal{H}_p) \right) \ ^{oldsymbol{N}} oldsymbol{P}_k &= (oldsymbol{I} - oldsymbol{K}_k oldsymbol{H}_k)^{oldsymbol{N}} ar{oldsymbol{P}}_k (oldsymbol{I} - oldsymbol{K}_k oldsymbol{H}_k)^T \end{aligned}$$



Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner

- 1. GetInput()
- 2. Motion Model
- 3. GetScan()
- 4. Observation Model
- 5. Registration

```
\operatorname{Method} Localization(\hat{x}_0, P_0)
     [{}^{N}\hat{x}_{0}, {}^{N}P_{0}] = [{}^{N}\hat{x}_{B_{0}}, {}^{N}P_{B_{0}}];
                                                                                                            // Initialize SLAM state vecto
     [B_0S_0, B_0R_{S_0}] = GetScan();;
                                                                                                           // Read the Scan from the sensor
     [^{N}\hat{\bar{x}}_{\mathbf{0}}, ^{N}P_{\mathbf{0}}] = AddNewPose(^{N}\hat{x}_{\mathbf{0}}, ^{N}\bar{P}_{\mathbf{0}});
                                                                                                                        // Grow the state vecto
     \mathcal{M} = [^{B_0}S_0];
                                                                                                                 // Store the scan in the m
     for k = 1 to steps do
     [u_k, Q_k] = GetInput();
     [^{N}\hat{\bar{x}}_{k}, ^{N}P_{k}] = Prediction(^{N}\hat{x}_{k-1}, ^{N}P_{k-1}, u_{k}, Q_{k});
           [z_m, R_m] = GetMeasurement();
                                                                                                                    // Read navigation sensor
          if ScanAvailable then
                [B_k S_k, B_k R_{S_k}] = GetScan();
                 [^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, ^{\mathbf{N}}\bar{P}_{\mathbf{k}}] = AddNewPose(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, ^{\mathbf{N}}\bar{P}_{\mathbf{k}});
                                                                                                                       // Grow the state vector
                \mathcal{M}[k] = {}^{B_k}S_k;
                                                                                                                // Store the scan in the may
                \mathcal{H}_p = OverlappingScans(^{\mathbf{N}}\hat{\bar{x}}_{\mathbf{k}}, \mathcal{M});
                                                                                                        // Get pairs of overlapping scans
                for i = 1 to length(\mathcal{H}_n) do
                                                                                                                               // for all overlaps
                     j = \mathcal{H}_p[i];
                                                                                                  // Get the pair: B_j S_j overlaps B_k S_k
                     {}^{B_j}S_i = \mathcal{M}[i]; {}^{B_k}S_k = \mathcal{M}[k];
                                                                                                               // Get the scans from the man
                    ^{B_j}x_{B_k} = (\ominus^N x_{B_i}) \oplus ^N x_{B_k};
                     [\mathbf{z}_n, \mathbf{R}_n] = Register(^{B_j}S_i, ^{B_k}S_k, ^{B_j}x_{B_i});
                                                                                                                                // Go to next pair
          \mathcal{H}_{p} = DataAssociation(^{N}\hat{\bar{x}}_{k}, ^{N}\bar{P}_{k}, z_{p}, R_{p});
                                                                                                       // select compatible registration
          [z_k, R_k, H_k, V_k] = ObservationMatrix(\mathcal{H}_p, ^N \hat{\bar{x}}_k, z_m, R_m, z_p, R_p);
           [^{N}\hat{x}_{k}, ^{N}P_{k}] = Update(^{N}\hat{x}_{k}, ^{N}\bar{P}_{k}, z_{k}, R_{k}, H_{k}, V_{k}, \mathcal{H}_{p});
```

- > To implement a PEKFSLAM we need to:
  - 1. Program the **GetInput(·)** function
  - 2. Define the motion model:  ${}^{N}ar{x}_{B_{k}} = f\left({}^{N}x_{B_{k-1}}, u_{k}, w_{k}\right)$ 
    - Compute the motion model Jacobians:

$$\boldsymbol{J_{f_{x}}} = \frac{\partial f\left(^{N}\boldsymbol{x_{B_{k-1}}}, \boldsymbol{u_{k}}, \boldsymbol{w_{k}}\right)}{\partial^{N}\boldsymbol{x_{B_{k-1}}}} \;,\; \boldsymbol{J_{f_{w}}} = \frac{\partial f\left(^{N}\boldsymbol{x_{B_{k-1}}}, \boldsymbol{u_{k}}, \boldsymbol{w_{k}}\right)}{\partial \boldsymbol{w_{k}}}$$

- 3. Program the GetScan() function
- 4. Define the observation model:

$$egin{aligned} egin{aligned} egin{aligned\\ egin{aligned} egi$$

• Compute the observation Jacobians:

 $\forall i, j \text{ compute:}$ 

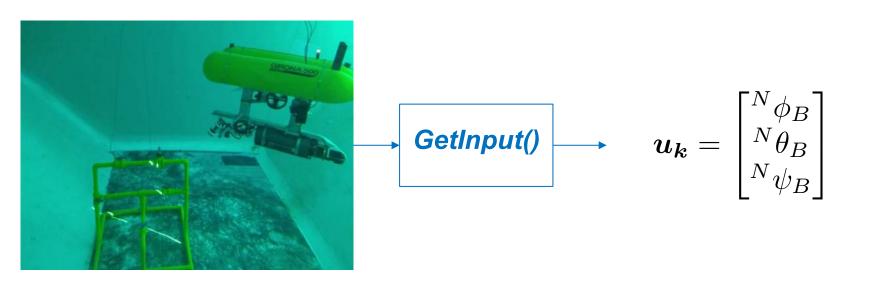
$$egin{aligned} oldsymbol{J_{h_x}} &= rac{\partial h_{ij}(^Nar{x}_k, v_k)}{\partial^N x_{B_t}} \;,\; oldsymbol{J_{h_v}} &= rac{\partial h_{ij}(^Nar{x}_k, v_k)}{\partial v_k} \end{aligned}$$

5. Implement the **Register(·)** function



Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner

1. GetInput() 2. Motion Model 3. GetScan() 4. Observation Model 5. Registration







1. GetInput()

2. Motion Model

3. GetScan()

4. Observation Model

5. Registration

#### **Motion Model**

State Vector:

$${}^{\boldsymbol{N}}\boldsymbol{x}_{\boldsymbol{B_k}} = [\underbrace{{}^{\boldsymbol{N}}\boldsymbol{x}_{\boldsymbol{B}} {}^{\boldsymbol{N}}\boldsymbol{y}_{\boldsymbol{B}} {}^{\boldsymbol{N}}\boldsymbol{z}_{\boldsymbol{B}}}_{\boldsymbol{N}\boldsymbol{\eta}_{\boldsymbol{1}_{\boldsymbol{B_k}}}} \underbrace{{}^{\boldsymbol{B}}\boldsymbol{u}_{\boldsymbol{B}} {}^{\boldsymbol{B}}\boldsymbol{v}_{\boldsymbol{B}} {}^{\boldsymbol{B}}\boldsymbol{w}_{\boldsymbol{B}}}_{\boldsymbol{B}\boldsymbol{\nu}_{\boldsymbol{A}}} \underbrace{{}^{\boldsymbol{N}}\boldsymbol{x}_{\boldsymbol{B_n}} \ldots {}^{\boldsymbol{N}}\boldsymbol{x}_{\boldsymbol{B_1}}]^T$$

Motion Model:

$$N_{\bar{\boldsymbol{x}}_{B_{k}}} = f\left(^{N}\boldsymbol{x}_{B_{k-1}}, \boldsymbol{u}_{k}, \boldsymbol{w}_{k}\right) = \begin{bmatrix} N_{\boldsymbol{\eta}_{1_{B_{k-1}}}} + N_{\boldsymbol{R}_{B}}(\boldsymbol{u}_{k} + \boldsymbol{w}_{\boldsymbol{\eta}_{2_{k}}}) \left(^{B}\boldsymbol{\nu}_{k-1}\Delta t + \boldsymbol{w}_{\dot{\boldsymbol{\nu}}_{k}}\frac{\Delta t^{2}}{2}\right) \\ B_{\boldsymbol{\nu}_{k-1}} + \boldsymbol{w}_{\dot{\boldsymbol{\nu}}_{k}}\Delta t \end{bmatrix}$$
where  $\boldsymbol{u}_{k} = \begin{bmatrix} N_{\boldsymbol{\phi}_{B}} & N_{\boldsymbol{\theta}_{B}} & N_{\boldsymbol{\psi}_{B}} \end{bmatrix}^{T}$ 

$$\boldsymbol{w}_{k} = \begin{bmatrix} \boldsymbol{w}_{\boldsymbol{\eta}_{2_{k}}}^{T} & \boldsymbol{w}_{\dot{\boldsymbol{\nu}}_{k}}^{T} \end{bmatrix}^{T} = \begin{bmatrix} w_{\boldsymbol{\phi}} & w_{\boldsymbol{\theta}} & w_{\boldsymbol{\psi}} & w_{\dot{u}} & w_{\dot{v}} & w_{\dot{w}} \end{bmatrix}^{T}$$

$$oldsymbol{w_k} = \mathcal{N}(\mathbf{0}, oldsymbol{Q_k}); oldsymbol{Q_k} = diag(\sigma_{w_\phi}^2, \sigma_{w_\theta}^2, \sigma_{w_\psi}^2, \sigma_{w_\psi}^2, \sigma_{w_\psi}^2, \sigma_{w_\psi}^2)$$

Jacobians:

$$egin{aligned} oldsymbol{J_{f_{w}}} &= rac{\partial f\left(^{N}x_{B_{k-1}}, u_{k}, w_{k}
ight)}{\partial^{N}x_{B_{k-1}}} = egin{bmatrix} oldsymbol{I_{3 imes3}} & ^{N}R_{B}(u_{k})\Delta t \ oldsymbol{0_{3 imes3}} & oldsymbol{I_{3 imes3}} \end{bmatrix} \ oldsymbol{J_{f_{w}}} &= rac{\partial f\left(^{N}x_{B_{k-1}}, u_{k}, w_{k}
ight)}{\partial w_{k}} = egin{bmatrix} oldsymbol{J_{R}} & ^{B}\hat{
u}_{k-1}\Delta t & ^{N}R_{B}(u_{k})rac{\Delta t^{2}}{2} \ oldsymbol{0_{3 imes3}} & oldsymbol{I_{3 imes3}}\Delta t \end{bmatrix} \ egin{bmatrix} oldsymbol{J_{R}} & ^{N}R_{B}(u_{k} + w_{\eta_{2_{k}}}) \end{bmatrix} \end{aligned}$$

where 
$$J_R=\left.rac{\partial^N R_B(u_k+w_{\eta_{2_k}})}{\partial w_{\eta_{2_k}}}
ight|_{N_{oldsymbol{X}_{B_{k-1}}}=^N\hat{oldsymbol{x}}_{B_{k-1}},w_k=0}$$





2. Motion Model

4. Observation Model 5. Registration

### View poses

- > The State Vector contains only the position:  ${}^N x_k = [\eta_{1_{B_k}}^T \ 
  u_1^T]^T$
- > A View Pose can be computed combining the position and the input attitude:

$$egin{aligned} egin{aligned} egin{aligned} N x_k &= [oldsymbol{\eta_{1_{B_k}}}^T oldsymbol{
u_1}^T oldsymbol{\eta_{2_k}} &= [oldsymbol{\eta_{1_{B_k}}} oldsymbol{\psi_1}^T]^T \ oldsymbol{u_k} &= oldsymbol{\eta_{2_{B_k}}} = egin{bmatrix} \phi & oldsymbol{ heta} & oldsymbol{\eta_1} \ oldsymbol{u_{1_k}} &= oldsymbol{I} \ oldsymbol{\eta_{2_i}} \end{bmatrix} = egin{bmatrix} oldsymbol{I} \ oldsymbol{\eta_{2_i}} \end{bmatrix} f(x_{k-1}, u_k, w_k) + oldsymbol{I} \ oldsymbol{I} \ oldsymbol{I} \end{bmatrix} (u_k + w_{oldsymbol{\eta_2}}) \ oldsymbol{w_{oldsymbol{\eta_2}}} &= oldsymbol{N} oldsymbol{I} \end{bmatrix} f(x_{k-1}, u_k, w_k) + oldsymbol{I} oldsymbol{I} \ oldsymbol{I} \end{bmatrix} (u_k + w_{oldsymbol{\eta_2}}) \end{aligned}$$

> Adding a new View Pose can be achieved by:

$$egin{aligned} ar{x}_k^+ = egin{bmatrix} Ff(x_{k-1}, u_k, w_k) + G(u_k + w_{oldsymbol{\eta_2}}) \ x_k \end{bmatrix}; F = egin{bmatrix} I \ 0 \end{bmatrix}; \ G = egin{bmatrix} 0 \ I \end{bmatrix} \end{aligned}$$



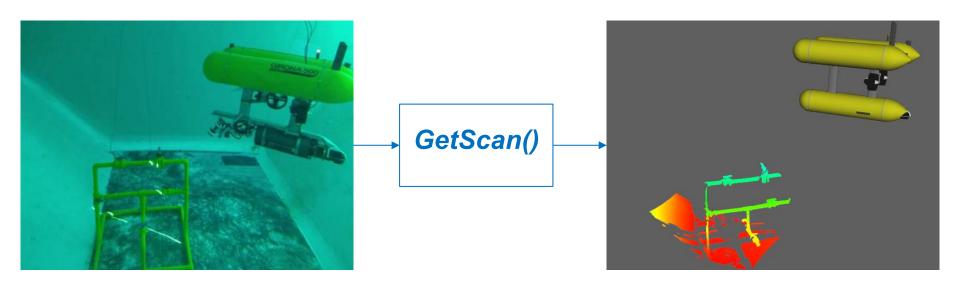


1. GetInput()

Motion Model

3. GetScan()

4. Observation Model



Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner

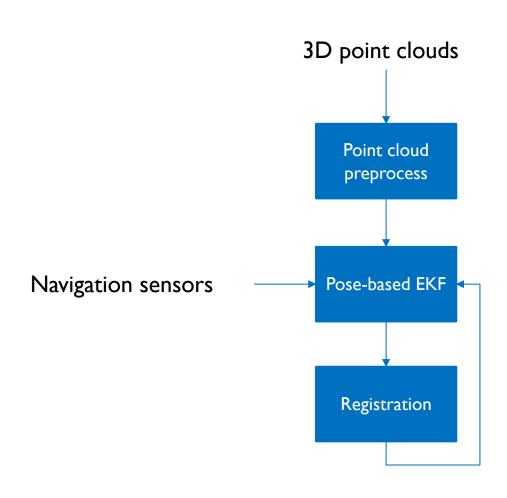


1. GetInput()

2. Motion Mode

3. GetScan()

4. Observation Model





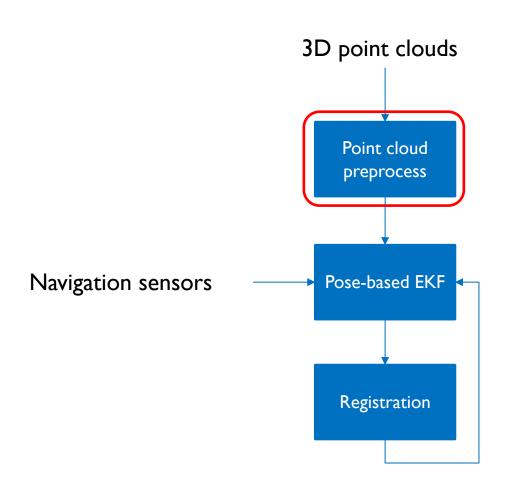


1. GetInput()

2. Motion Mode

GetScan()

4. Observation Model



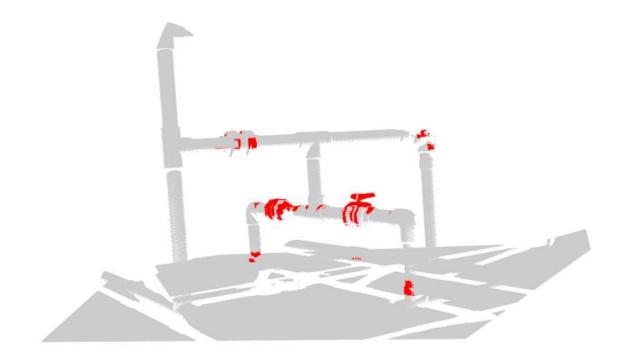
Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner



4. Observation Model 5. Registration

### **Point cloud preprocess**

- Key points extraction
  - Remove planar surfaces (RANSAC, Fischler et al. 1981)
  - Remove points with curvature (Pauly et al., 2002) below threshold



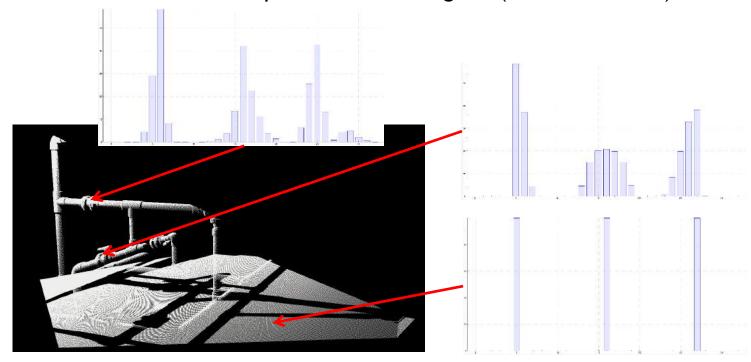




- 1. GetInput()
- 2. Motion Mode
- 3. GetScan()
- 4. Observation Model
- Registration

## **Point cloud preprocess**

- Key points extraction
  - Remove planar surfaces (RANSAC, Fischler et al. 1981)
  - Remove points with curvature (Pauly et al., 2002) below threshold
- Feature extraction: Fast point feature histogram (Rusu et al., 2009)



Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner



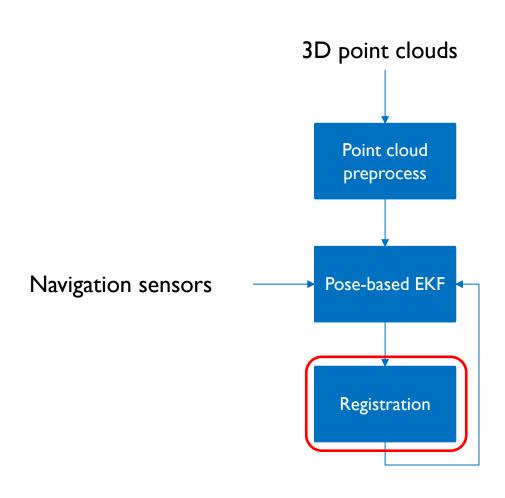
1. GetInput()

Motion Model

3. GetScan()

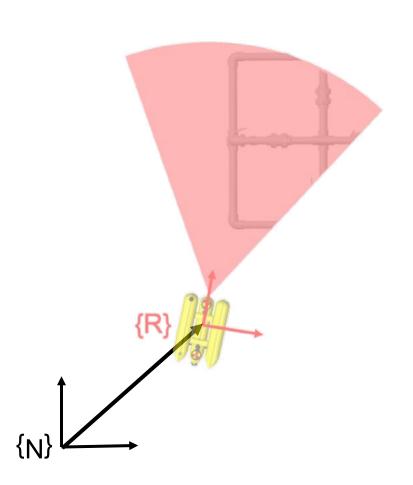
4. Observation Model

Registration





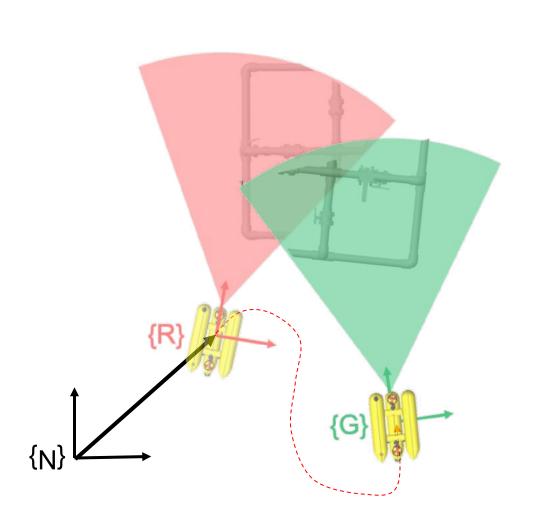
> The AUV takes 1 scans and adds the View-Pose to the map



$$\mathcal{M} = [^R S_R]$$
  $^N \hat{x}_k = egin{bmatrix} ^N \hat{x}_R \ ^N \hat{ar{x}}_{B_k} \end{bmatrix}$ 



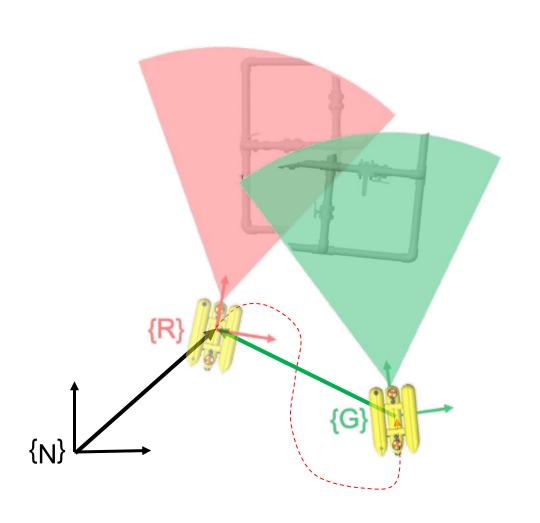
> The AUV moves and take another scans from the new View pose



$$\mathcal{M} = [^{m{R}} m{S_R}, \ ^{m{G}} m{S_G}]$$
  $^{m{N}} \hat{ar{x}}_{m{k}} = egin{bmatrix} ^{m{N}} \hat{m{x}}_{m{R}} \ ^{m{N}} \hat{m{x}}_{m{B_k}} \end{bmatrix}$ 



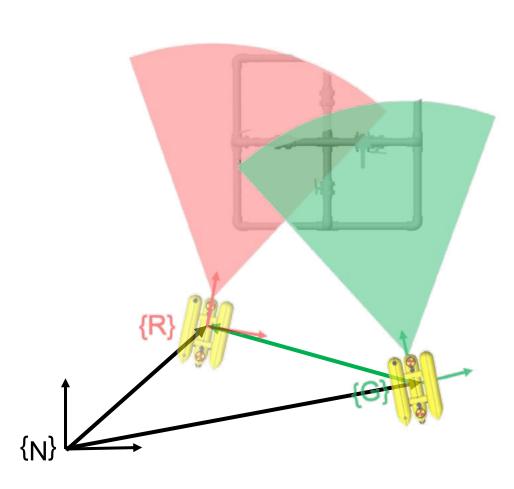
> The Observation function computes the expected robot displacement between scans.



$$egin{aligned} \mathcal{M} &= [^RS_R,\ ^GS_G] \ &^N\hat{ar{x}}_k = \left[ egin{aligned} ^N\hat{x}_R \ ^N\hat{x}_G \ ^N\hat{ar{x}}_{B_k} \end{aligned} 
ight] \ & oldsymbol{z_{RG}} &= oldsymbol{h_{RG}}(^Nx_k,v_k) = \ominus^Nx_R \oplus ^Nx_G + v_{RG} \end{aligned}$$



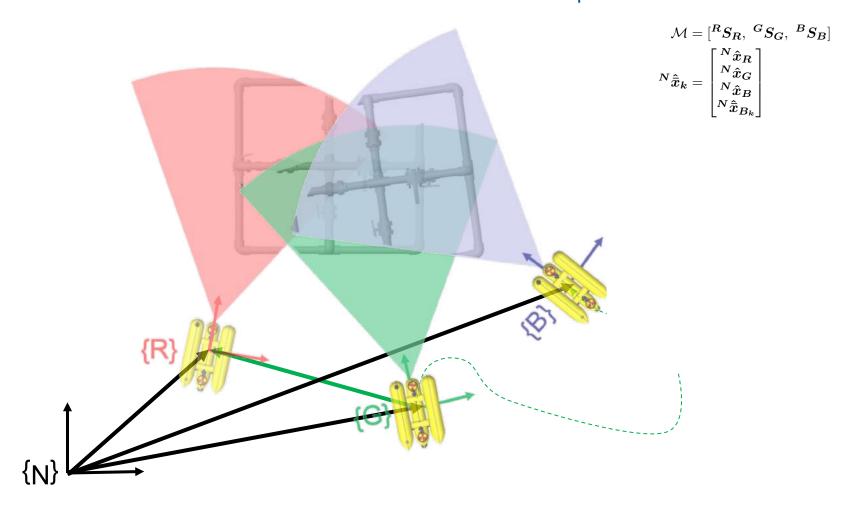
> Scans are registered to get the robot displacement & the PEKFSLAM is updated



$$\mathcal{M} = [^RS_R, \ ^GS_G]$$
 $N\hat{x}_k = \begin{bmatrix} N\hat{x}_R \\ N\hat{x}_G \\ N\hat{x}_{B_k} \end{bmatrix}$ 
 $z_{RG} = h_{RG}(^Nx_k, v_k) = \ominus^Nx_R \oplus^Nx_G + v_{RG}$ 
 $[z_{RG}, R_{RG}] = Register(^RS_R, ^GS_G)$ 
 $N\hat{x}_k = \begin{bmatrix} N\hat{x}_G \\ N\hat{x}_R \\ N\hat{x}_{B_k} \end{bmatrix} = Update(\cdot)$ 

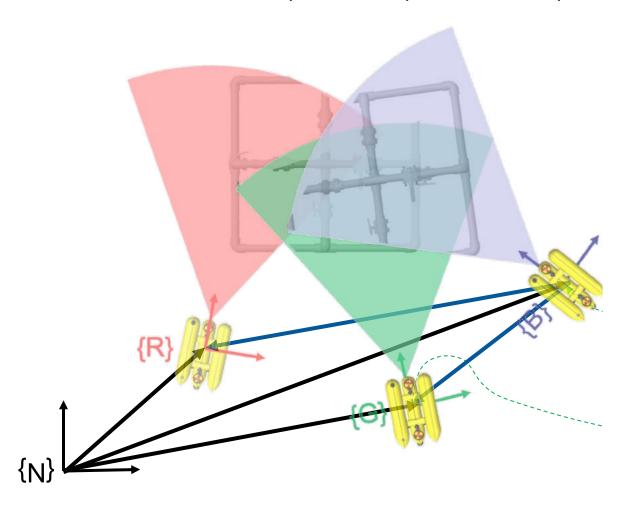


> The AUV moves and take another scans from the new View pose





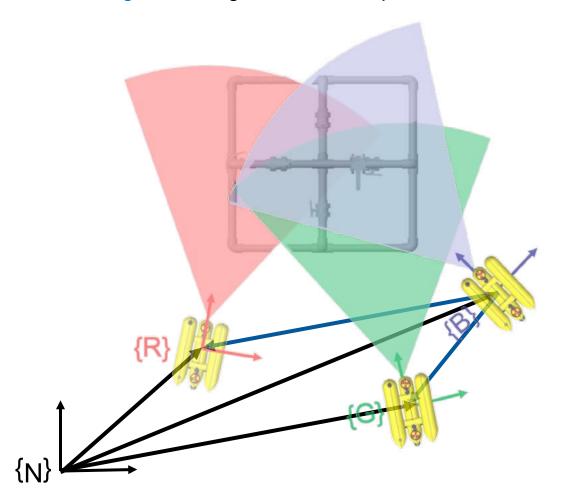
> The Observation function computes the expected robot displacement between scans.



$$egin{aligned} \mathcal{M} &= [^RS_R,\ ^GS_G,\ ^BS_B] \ ^N\hat{ar{x}}_k &= \begin{bmatrix} ^N\hat{x}_R \ ^N\hat{x}_G \ ^N\hat{x}_B \ ^N\hat{ar{x}}_{B_k} \end{bmatrix} \ z_k &= egin{bmatrix} h_{BG}(^Nx_k,v_k) \ h_{BR}(^Nx_k,v_k) \end{bmatrix} \end{aligned}$$



> Scans are registered to get the robot displacement & the PEKFSLAM is updated



$$\mathcal{M} = \begin{bmatrix} ^RS_R, \ ^GS_G, \ ^BS_B \end{bmatrix}$$
 $N\hat{x}_k = \begin{bmatrix} ^N\hat{x}_R \\ ^N\hat{x}_G \\ ^N\hat{x}_B \\ ^N\hat{x}_{B_k} \end{bmatrix}$ 
 $z_k = \begin{bmatrix} h_{BG}(^Nx_k, v_k) \\ h_{BR}(^Nx_k, v_k) \end{bmatrix}$ 
 $= \begin{bmatrix} \ominus^Nx_B \oplus ^Nx_G + v_{BG} \\ \ominus^Nx_B \oplus ^Nx_R + v_{BR} \end{bmatrix}$ 
 $[z_{BG}, R_{BG}] = Register(^BS_B, ^GS_G)$ 
 $[z_{BR}, R_{BR}] = Register(^BS_B, ^RS_R)$ 
 $z_k = \begin{bmatrix} z_{BG} \\ z_{BR} \end{bmatrix}; \ R_k = \begin{bmatrix} R_{BG} & 0 \\ 0 & R_{BR} \end{bmatrix}$ 
 $N\hat{x}_k = \begin{bmatrix} ^N\hat{x}_G \\ ^N\hat{x}_B \\ ^N\hat{x}_{B_k} \end{bmatrix} = Update(\dots, z_k, R_k)$ 





1. GetInput()

Motion Mode

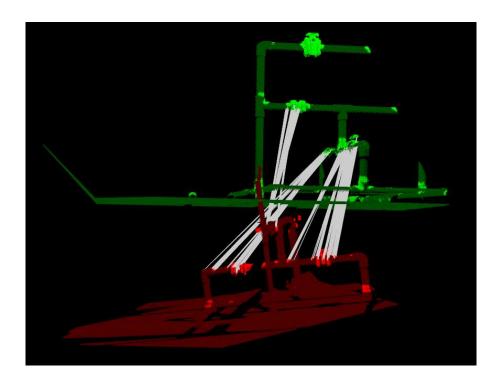
3. GetScan()

4. Observation Mode

5. Registration

## Registration algorithm

- Coarse registration
  - Feature association
  - Roto-translation using Singular Value Decomposition (J. Besl and McKay, 1992)



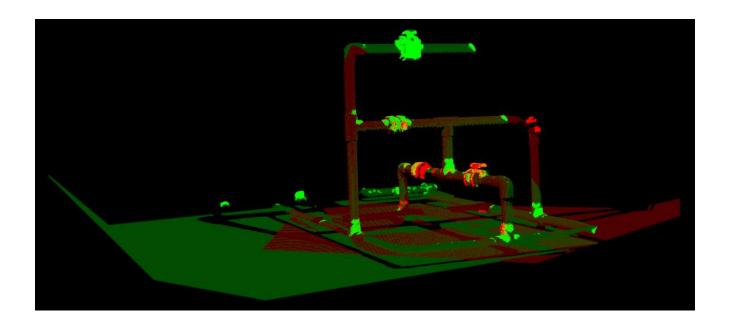




4. Observation Model 5. Registration

## Registration algorithm

- Coarse registration
  - Feature association
  - Roto-translation using Singular Value Decomposition (J. Besl and McKay, 1992)
- Fine registration: Point to point ICP

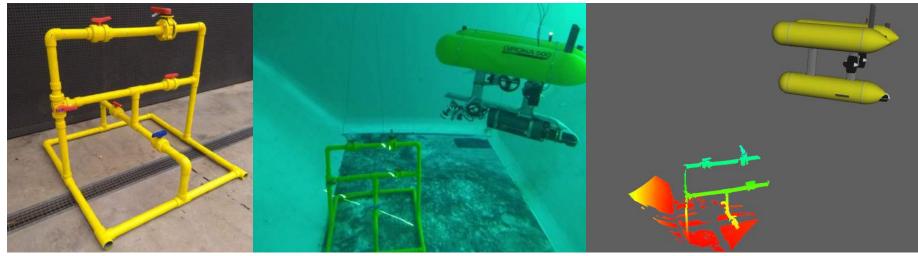


Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner



# **Experiments and results**

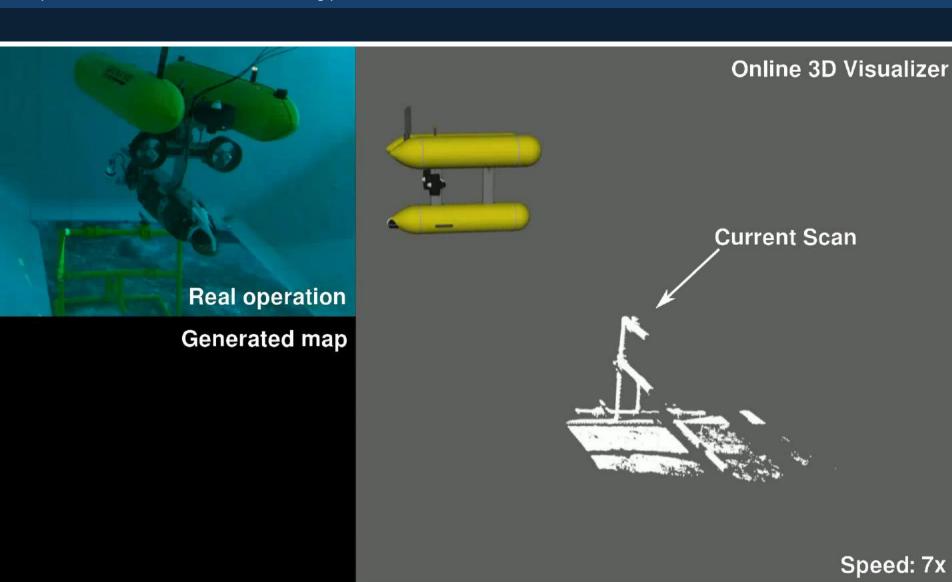




Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner



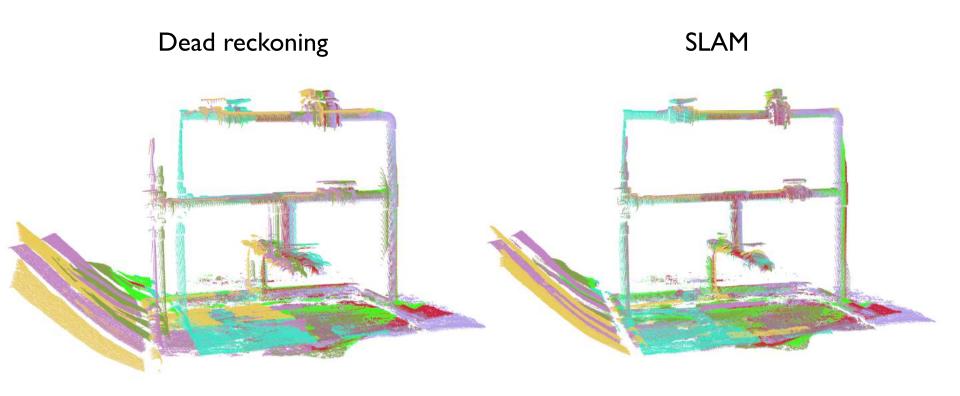
Speed: 7x







# **Experiments and results**



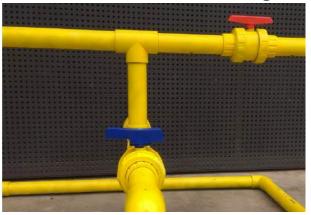




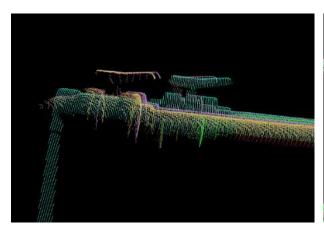
## **Experiments and results**

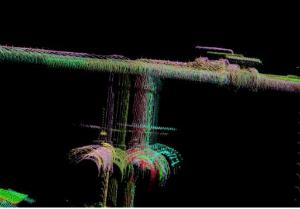
Structure details: dead reckoning navigation

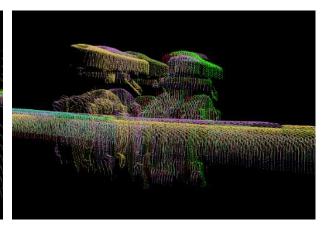












Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner



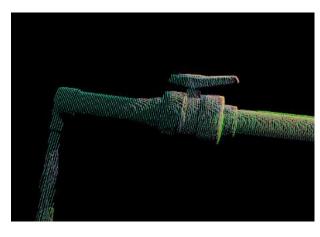
## **Experiments and results**

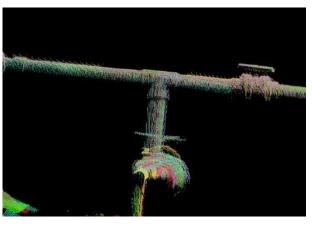
Structure details: SLAM















## **Experiments and results**

## Robot uncertainty

