# SLIDING WINDOW FILTER BASED UNKNOWN OBJECT POSE ESTIMATION

UNIVERSITY OF TORONTO

{ JIARU SONG }   UNIVERSITY OF TORONTO INSTITUTE OF AEROSPACE STUDIES

Institute for Aerospace Studies UNIVERSITY OF TORONTO

## OVERVIEW

We propose a novel framework for unknown object pose estimation in a Sliding Window Filter(SWF) manner with the following object properties :

1. No prior information of shape and size are available
2. No motion assumption is required
3. No sensors are equipped on the object

The structure and pose of object on SE(3) are estimated simultaneously. Gauss-Newton (GN) method is implemented for each window with an initial guess generated by OPnP[1] algorithm.

## PROBLEM SETUP

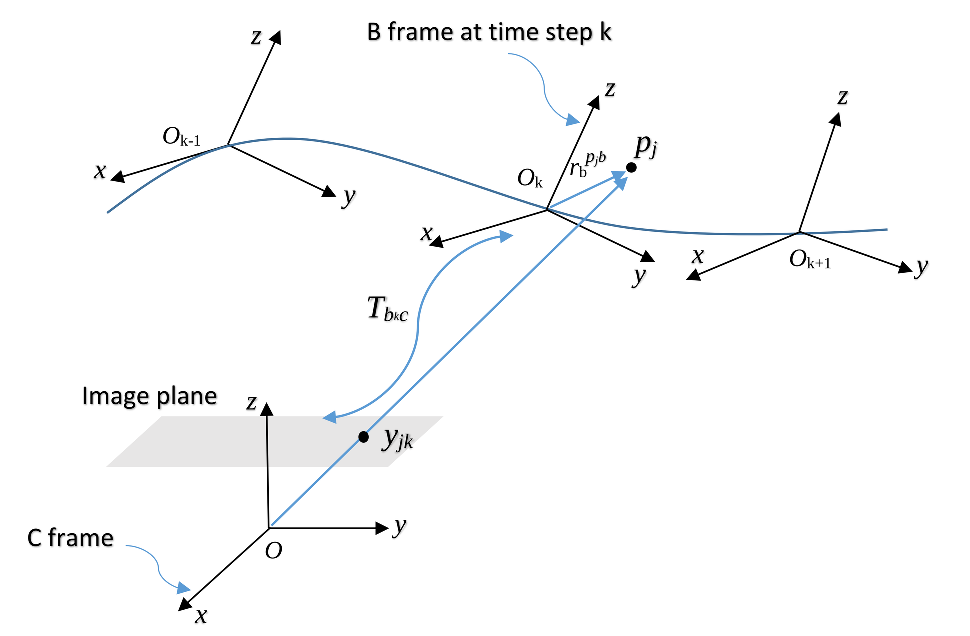


**Figure 1:** Problem Setup

As shown in Fig.1, the estimated states are pose $\boldsymbol{T}_k$ and structure of the feature point $\boldsymbol{p}_j$:

$$\boldsymbol{T}_k = \boldsymbol{T}_{c,b_k} = \begin{bmatrix} \boldsymbol{R}_{c,b_k} & \boldsymbol{t}_c^{b_k,c} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \quad \boldsymbol{p}_j = \begin{bmatrix} \boldsymbol{r}_b^{p_j b} \\ 1 \end{bmatrix} \tag{1}$$

where $k = 1, ..., K$, $K$ is the window size of SWF; $j = 1, ..., M$, $M$ is the total number of visible feature points of one window. The measurements $y_{jk}$, which are corrupted by zero mean Gaussian noise $\boldsymbol{n}_{jk}$, are pixel coordinates$(u_{jk}, v_{jk})$ and depth $d_{jk}$ from RGB-D camera. The observation model is

$$\boldsymbol{y}_{jk} = \boldsymbol{\pi}(\boldsymbol{x}_{jk}) + \boldsymbol{n}_{jk}, \boldsymbol{n}_{jk} \sim N(\boldsymbol{0}, \boldsymbol{Q}_{jk}) \tag{2}$$

with the following shorthand: $\boldsymbol{x} = \{\boldsymbol{T}_1, ..., \boldsymbol{T}_K, \boldsymbol{p}_1, ..., \boldsymbol{p}_M\}$, as well as $\boldsymbol{x}_{jk} = \{\boldsymbol{T}_k, \boldsymbol{p}_j\}$ for $k$th pose and $j$th feature point.

## POSE ESTIMATION ON SE(3)

The measurement inconsistency error of $j$th feature point from $k$th pose of a sliding window is defined as:

$$\boldsymbol{e}_{y,jk}(\boldsymbol{x}) = \boldsymbol{y}_{jk} - \boldsymbol{\pi}(\boldsymbol{x}_{jk}) \tag{3}$$

where $\boldsymbol{z}(\boldsymbol{\pi}_{jk})$ transforms feature coordinates from object frame to camera frame, and then projects the coordinates from camera frame into image plane. The objective function is to minimize the sum of squared inconsistency error given states $\boldsymbol{x}$:

$$J(\boldsymbol{x}) = \frac{1}{2} \sum_{j,k} \boldsymbol{e}_{y,jk}(\boldsymbol{x})^T \boldsymbol{Q}_{jk}^{-1} \boldsymbol{e}_{y,jk}(\boldsymbol{x}) \tag{4}$$

The states at operating points are perturbed by:

$$\begin{aligned} \boldsymbol{T}_k &= \exp(\boldsymbol{\epsilon}_k^\wedge) \boldsymbol{T}_{op,k} \approx (\boldsymbol{1} + \boldsymbol{\epsilon}_k^\wedge) \boldsymbol{T}_{op,k} \\ \boldsymbol{p}_j &= \boldsymbol{p}_{op,j} + \boldsymbol{D} \boldsymbol{\zeta}_j \end{aligned} \tag{5}$$

where operator $^\wedge$ is defined as

$$\epsilon_{6\times1}^\wedge = \begin{bmatrix} \rho_{3\times1} \\ \phi_{3\times1} \end{bmatrix}^\wedge = \begin{bmatrix} \phi^\times & \rho_{3\times1} \\ 0_{1\times3} & 0 \end{bmatrix}_{4\times4} \tag{6}$$

$\times$ is infinitesimal rotation and $\exp$ is exponential forward mapping on SE(3). To implement GN method, the observation model is linearized as:

$$\boldsymbol{\pi}(\boldsymbol{x}_{jk}) \approx \boldsymbol{\pi}(\boldsymbol{x}_{op,jk}) + \boldsymbol{\Pi}_{jk} \boldsymbol{\delta x}_{jk} \tag{7}$$

where $\boldsymbol{\Pi}_{jk}$ is the Jacobian of function $\boldsymbol{\pi}$ with respect to perturbed states. Thus, the objective function can then be rearranged as:

$$J(\boldsymbol{x}) \approx J(\boldsymbol{x}_{op}) - \mathbf{b}^T \delta \boldsymbol{x} + \frac{1}{2} \delta \boldsymbol{x}^T \boldsymbol{A} \delta \boldsymbol{x} \tag{8}$$

$$\begin{aligned} \boldsymbol{b} &= \boldsymbol{H}^T \boldsymbol{Q}^{-1} \boldsymbol{e}(\boldsymbol{x}_{op}) \\ \boldsymbol{A} &= \boldsymbol{H}^T \boldsymbol{Q}^{-1} \boldsymbol{H} \end{aligned} \tag{9}$$

$$\begin{aligned} \boldsymbol{H} &= \left[\boldsymbol{H}_{10}^T, ..., \boldsymbol{H}_{M0}^T, \boldsymbol{H}_{11}^T, ..., \boldsymbol{H}_{MK}^T\right]^T \\ \boldsymbol{Q} &= \mathrm{diag}\{\boldsymbol{Q}_{10}, ..., \boldsymbol{Q}_{M0}, \boldsymbol{Q}_{11}, ..., \boldsymbol{Q}_{MK}\} \\ \boldsymbol{e}(\boldsymbol{x}_{op}) &= \left[\boldsymbol{e}_{y,10}(\boldsymbol{x}_{op})^T, ..., \boldsymbol{e}_{y,MK}(\boldsymbol{x}_{op})^T\right]^T \\ \boldsymbol{H}_{jk} &= \boldsymbol{\Pi}_{jk} \boldsymbol{P}_{jk} \end{aligned} \tag{10}$$

$\boldsymbol{P}_{jk}$ is a projection matrix to select the visible $j$th measurement at time step $k$ of the overall perturbed state. The minimum value of objective function is then calculated by:

$$\boldsymbol{A} \boldsymbol{\delta x}^* = \boldsymbol{b} \tag{11}$$

Then the operating points are updated by:

$$\begin{aligned} \boldsymbol{T}_{op,k} &\leftarrow \exp(\boldsymbol{\epsilon}_k^{*\wedge}) \boldsymbol{T}_{op,k} \\ \boldsymbol{p}_{op,j} &\leftarrow \boldsymbol{p}_{op,j} + \boldsymbol{D} \boldsymbol{\zeta}^*{}_j \end{aligned} \tag{12}$$

When SWF with window size $K$ slides along the time line, the above algorithm runs iteratively for each window until convergence. In the first window, the overall estimated states will be recorded with initial state $\boldsymbol{T}_{b_0 c}$ and $\boldsymbol{p}_{j0}$ generated by Schmidt Orthogonalization. For the rest of windows, the initial state of window $l$ is the first estimated state in window $l-1$ with initial guess generated by OPnP[1] based algorithm, as summarized follows:

**Algorithm 1** Initial Guess Generation

1: Match the feature points of time step $k-1$ and time step $k$. Set the 3D position $\boldsymbol{p}_{j(k-1)}$ of matched feature points as the initial guess $\check{\boldsymbol{p}}_{jk1}$;
2: Use the initial guess $\check{\boldsymbol{p}}_{jk}$ and measurement $\boldsymbol{y}_{jk}$ to solve the initial pose $\check{\boldsymbol{T}}_{b_k c}$ by OPnP; when the measurement is insufficient $\check{\boldsymbol{T}}_{b_k c} = \boldsymbol{T}_{b_{(k-1)} c}$;
3: Use observation model and initial pose $\check{\boldsymbol{T}}_{b_k c}$ to compute 3D position $\check{\boldsymbol{p}}_{jk2}$ of unmatched feature points. Combine the $\check{\boldsymbol{p}}_{jk1}$ and $\check{\boldsymbol{p}}_{jk2}$ together to form initial points $\check{\boldsymbol{p}}_{jk}$;

## CONCLUSION

The experiment shows that the proposed SWF-based framework can estimate the pose of a unknown object with different shape and arbitrary trajectory accurately. Besides, the framework is robust to the number of feature point within each window compared with OPnP algorithm.

## REFERENCES

[1] Yinqiang Zheng, Yubin Kuang, Shigeki Sugimoto, Kalle Astrom, and Masatoshi Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013.

## RESULTS

The pose error is defined as translational error and rotational error with estimated value $x^*$:

$$\begin{aligned} \delta \boldsymbol{r}_k &= \begin{bmatrix} \delta r_{x,k} & \delta r_{y,k} & \delta r_{z,k} \end{bmatrix}^T := \bar{\boldsymbol{r}}_c^{b_k c^*} - \boldsymbol{r}_c^{b_k c} \\ \delta \boldsymbol{\theta}_k^\times &= \begin{bmatrix} \delta\theta_{x,k} \\ \delta\theta_{y,k} \\ \delta\theta_{z,k} \end{bmatrix}^\times := \boldsymbol{1} - \bar{\boldsymbol{R}}_{b_k c}^* \boldsymbol{R}_{b_k c}^T \end{aligned} \tag{13}$$

The objects, estimated errors and estimated trajectories are shown in Fig.2.

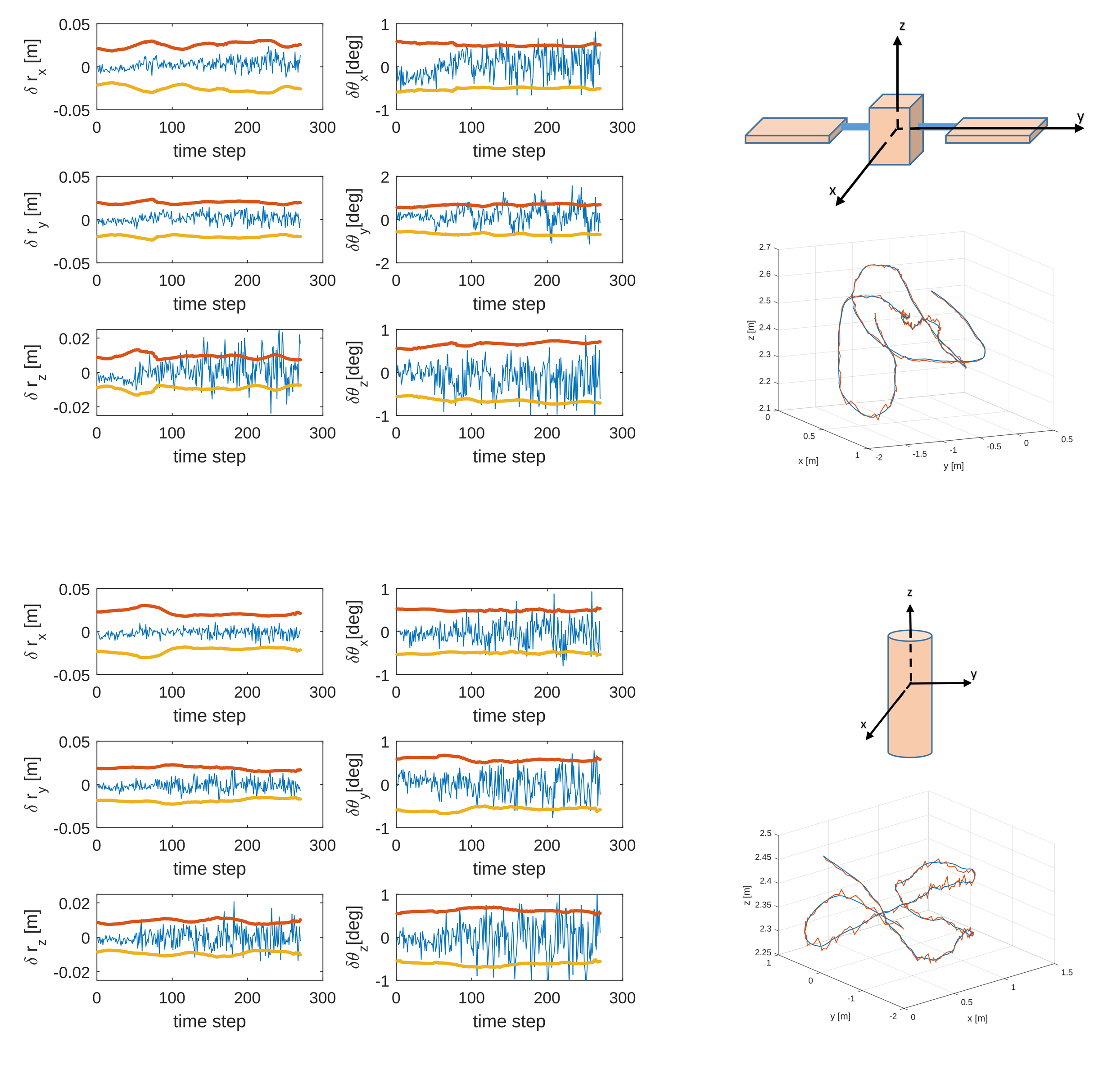


**Figure 2:** Results of two ojbects

To compare with OPnP in cases of different feature point, the following errors are defined as follows and Fig.3 shows the comparison result.

$$e_{rk} = \|\delta \boldsymbol{r}_k\|_2, \quad e_{\theta k} = \|\delta \boldsymbol{\theta}_k\|_2, \tag{14}$$

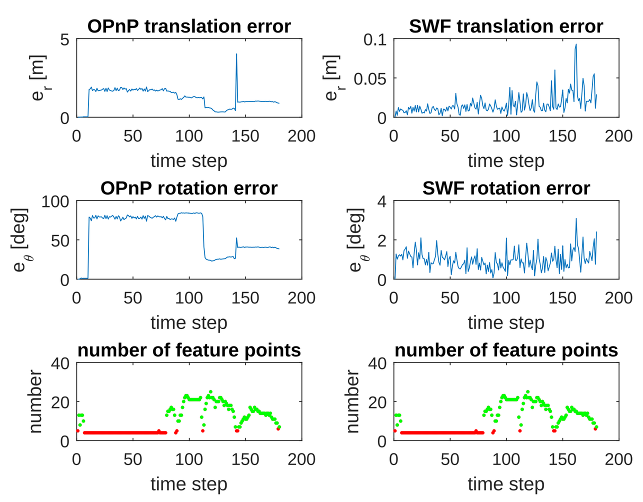


**Figure 3:** Comparison of OPnP