SLIDING WINDOW FILTER BASED UNKNOWN OBJECT POSE ESTIMATION

Jiaru Song

University of Toronto Institute of Aerospace Studies
4925 Dufferin St, North York, ON, Canada, M3H5T6

ABSTRACT

This paper proposes a novel framework for unknown object pose estimation. There are no sensors or prior information about the object. In order to explore the full current information and to achieve the online performance, smoothing technique is adopted by using sliding window filter (SWF) to estimate the structure and pose on SE(3) simultaneously. In addition, Gauss-Newton (GN) method is implemented for each window with an initial guess generated by OPnP algorithm. Unlike the existed smoothing algorithm, the proposed one is free of large estimation error and singular coefficient matrix. The experiment shows that the proposed framework is capable of estimating the pose of any arbitrary unknown object with any complex motion trajectory accurately even when the measured feature points are insufficient during object motion.

Index Terms— Pose Estimation, Sliding-Window Filter, Lie Group, Gauss-Newton Optimization

1. INTRODUCTION

The pose estimation of an unknown object is one of the most important problems in computer vision, Augmented Reality (AR), space applications, and robotics [1, 2, 3]. Object pose estimation with prior information is well studied. However, the pose estimation for unknown objects is more challenging: the prior information such as shape, size and initial pose is unknown and no sensors can be equipped on the object. In the past few years, several works on unknown object pose estimation have been published.

Unknown object pose is estimated mainly by camera based systems and Light Detection and Ranging (LIDAR) based systems. The former system, which uses RGB-D camera, monocular camera, and stereo camera, is prevalent in 3D object tracking in computer vision and AR [4, 5, 6, 7]; LIDAR is prevalent for space applications [8, 9]. Among these systems, there are numerous proposed methods based on the accessibility of point correspondences. When the 2D-3D point correspondences are unavailable, methods such as Truncated Signed Distance Function (TSDF)-based method and contour-based method have been proposed [4, 10]. However, the TSDF-based method is incapable of recovering the fine details of the scene and makes the pose estimation inaccurate as TSDF represents the alignment errors; contour-based method is based on two presumptions of the extracted contour, which is hard to achieve for unknown object. Although the Graph Cut is proposed for contour extraction in pose estimation, it depends on known ellipse feature [11].

When 2D-3D point correspondences are obtained, poses can be estimated by point-cloud based methods. The essence is to build a point-cloud model by using extracted feature points and depth information, and to minimize the reprojection errors, or object space linearity errors. SIFT, SURF features are commonly adopted [12, 13]; Viewpoint Feature Histogram (VFH) and Clustered Viewpoint Feature Histogram (CVFH) are derived for pose estimation using depth image [14, 15]. Pose from Orthography and Scaling with Iterations (POSIT), Orthogonal Iteration (OI), and Direct Linear Transformation (DLT) are proposed and widely used in pose estimation [16, 17, 18]. OI studies the optimization structure of pose estimation, and minimizes the reprojection error by converting the objective function of both rotation matrix and translation into the function of rotation matrix. Iterative Closest Point (ICP) is prevalent for LIDAR-based system when 3D-3D correspondences are available, but it requires a good initial guess. Another group of methods stem from Perspective-n-Points (PnP) problem. Methods such as EPnP, OPnP, and REPPnP are widely used in unknown pose estimation [19, 20, 21]. OPnP is an effective method in the cases where Gaussian noise is present; REPPnP is robust in the presence of outliers.

The methods mentioned above mainly focus on the estimation by using single-frame knowledge. However, sensors have noise, and there may be insufficient feature points for pose estimation in some time steps. Therefore, enhancing the efficacy of single-frame methods by making use of filtering or smoothing technique is necessary.

Because of the nonlinear motion model and observation model, the prevalent filtering methods are Extended Kalman Filter (EKF) and Iterated Extended Kalman Filter (IEKF) [1, 3, 5, 7, 22, 23]. The filter can be initialized by single-frame methods such as ICP and PnP algorithm [5, 23]. However, filtering methods are structureless, which cannot explore the current full information [24]. On the other hand, smoothing technique, making use of object structure, is more accurate than filtering approaches in nonlinear cases. The general practice is to form an optimization problem solved by GN in a window of recent states. One way to practice the smoothing technique is generating the structure by TSDF-model and then minimizing the sum of squared point-to-model distances from two consecutive images [4]. However, the method is still unable to explore the current information comprehensively, as the structure is not included in the estimated state.

This paper proposes a novel framework to estimate both the pose and structure of an unknown arbitrary object simultaneously without any prior information about the object using a RGB-D camera. In order to explore the full current information and to achieve the online performance, smoothing technique is adopted by using SWF to estimate the structure and pose on SE(3). In addition, GN is implemented for each window with an initial guess generated by OPnP [20]. Unlike the existed smoothing algorithm, the proposed one is free of large estimation error and singular coefficient matrix in [4]. The experiment tested estimation algorithms given two different objects moving along complicated trajectories. The efficacy and robustness of the framework are justified.

2. PROBLEM FORMATION

2.1. Problem setup

As shown in Fig.1, the object body frame is $\boldsymbol{B}$, which is fixed on the object. Camera frame is $\boldsymbol{C}$ with optical center as the origin and the optical axis as z. The states are pose $\boldsymbol{T}_k$ and structure of the feature point $\boldsymbol{p}_j$:

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

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 initial pose $\boldsymbol{T}_0$ is calculated before the SWF is started, and thus is excluded from system states. The measurements y_{jk} are noised pixel coordinates and depth of jth feature points of time step k. Measurements are obtained by a RGB-D camera corrupted by zero mean Gaussian noise $\boldsymbol{n}_{jk}$. The following shorthand is used:

$$\boldsymbol{x} = \{\boldsymbol{T}_1, ..., \boldsymbol{T}_K, \boldsymbol{p}_1, ..., \boldsymbol{p}_M\} \tag{2}$$

as well as $\boldsymbol{x}_{jk} = \{\boldsymbol{T}_k, \boldsymbol{p}_j\}$. Throughout the paper, subscript jk denotes the kth pose and jth feature point.

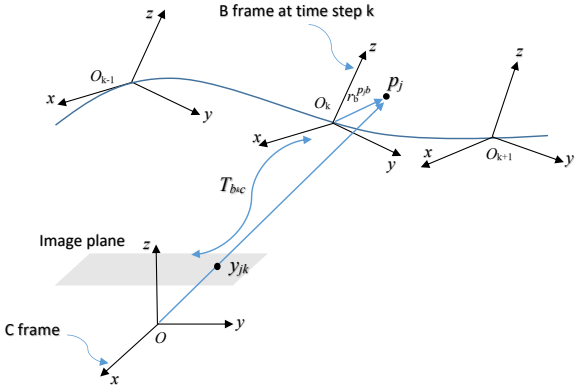


Fig. 1. Problem Setup

2.2. Observation model

The observation model is

$$\boldsymbol{y}_{jk} = \boldsymbol{\pi}(\boldsymbol{x}_{jk}) + \boldsymbol{n}_{jk}, \boldsymbol{n}_{jk} \sim N(\boldsymbol{0}, \boldsymbol{Q}_{jk}) \tag{3}$$

$\boldsymbol{\pi}(\boldsymbol{x}_{jk})$ can be written as two sub-functions:

$$\boldsymbol{\pi}(\boldsymbol{x}_{jk}) = \boldsymbol{s}(\boldsymbol{z}(\boldsymbol{x}_{jk})) \tag{4}$$

where

$$\boldsymbol{z}(\boldsymbol{x}_{jk}) = \boldsymbol{D}^{\mathrm{T}} \boldsymbol{T}_k \boldsymbol{p}_j \tag{5}$$

$$\boldsymbol{s}(\boldsymbol{z}(\boldsymbol{x}_{jk})) = \begin{bmatrix} u_{jk} & v_{jk} & d_{jk} \end{bmatrix}^{\mathrm{T}} \tag{6}$$

$$\begin{bmatrix} u_{jk} \\ v_{jk} \end{bmatrix} = \boldsymbol{F}\boldsymbol{K}\frac{1}{z_{jk,3}}\boldsymbol{z}_{jk}, d_{jk} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \boldsymbol{z}_{jk} \tag{7}$$

with $\boldsymbol{z}_{jk} = \boldsymbol{z}(\boldsymbol{x}_{jk}) = [z_{jk,1} \quad z_{jk,2} \quad z_{jk,3}]^{\mathrm{T}}$. $\boldsymbol{1}_n$ is a $n \times n$ identity matrix. $\boldsymbol{D}$, $\boldsymbol{F}$, and $\boldsymbol{K}$ are defined as follows:

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{1}_3 \\ \boldsymbol{0}_{1\times 3} \end{bmatrix}, \boldsymbol{F} = \begin{bmatrix} \boldsymbol{1}_2 & \boldsymbol{0}_{2\times 1} \end{bmatrix}, \boldsymbol{K} = \begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

the matrix $\boldsymbol{K}$ is the intrinsic calibration matrix of camera.

3. SLIDING WINDOW FILTER ON SE(3)

3.1. GN optimization

The reprojection error of jth feature point from kth pose of the sliding window is defined as:

$$\boldsymbol{e}_{y,jk}(\boldsymbol{x}) = \boldsymbol{y}_{jk} - \boldsymbol{\pi}(\boldsymbol{x}_{jk}) \tag{9}$$

where $k = 0, 1, ..., K$. Accordingly, the objective function is to minimize the sum of reprojection error within a window given states both pose and structure:

$$J(\boldsymbol{x}) = \frac{1}{2}\sum_{j,k} \boldsymbol{e}_{y,jk}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{Q}_{jk}^{-1} \boldsymbol{e}_{y,jk}(\boldsymbol{x}) \tag{10}$$

In order to implement GN method on SE(3), the perturbed states have been used to linearize the observation model.
The states at operating points are perturbed by:

$$\begin{aligned} \boldsymbol{T}_k &= \exp(\boldsymbol{\epsilon}_k^{\wedge})\boldsymbol{T}_{op,k} \approx (\boldsymbol{1} + \boldsymbol{\epsilon}_k^{\wedge})\boldsymbol{T}_{op,k} \\ \boldsymbol{p}_j &= \boldsymbol{p}_{op,j} + \boldsymbol{D}\boldsymbol{\zeta}_j \end{aligned} \tag{11}$$

where operator $^{\wedge}$ is defined as

$$\epsilon^{\wedge} = \begin{bmatrix} \rho \\ \phi \end{bmatrix}^{\wedge} = \begin{bmatrix} \phi^{\times} & \rho \\ 0_{1\times 3} & 0 \end{bmatrix} \tag{12}$$

where $\times$ is infinitesimal rotation and exp is exponential forward mapping. The following shorthands are used for the operating point and state perturbation:

$$\boldsymbol{x}_{op} = \{\boldsymbol{T}_{op,1}, ..., \boldsymbol{T}_{op,K}, \boldsymbol{p}_{op,1}, ..., \boldsymbol{p}_{op,M}\} \tag{13}$$

$$\delta\boldsymbol{x} = \left[\boldsymbol{\epsilon}_1^{\mathrm{T}} \cdots \boldsymbol{\epsilon}_K^{\mathrm{T}}, \boldsymbol{\zeta}_1^{\mathrm{T}} \cdots, \boldsymbol{\zeta}_M^{\mathrm{T}}\right]^{\mathrm{T}} \tag{14}$$

Referring to literature [25], the observation model is linearized as:

$$\begin{aligned} \boldsymbol{z}(\boldsymbol{x}_{jk}) =& \boldsymbol{D}^{\mathrm{T}}\boldsymbol{T}_k\boldsymbol{p}_j \\ \approx& \boldsymbol{D}^{\mathrm{T}}(\boldsymbol{1} + \boldsymbol{\epsilon}_k^{\wedge})\boldsymbol{T}_{op,k}(\boldsymbol{p}_{op,j} + \boldsymbol{D}\boldsymbol{\zeta}_j) \\ \approx& \boldsymbol{z}(\boldsymbol{x}_{op,jk}) + \boldsymbol{Z}_{jk}\boldsymbol{x}_{jk}\boldsymbol{\delta}\boldsymbol{x}_{jk} \end{aligned} \tag{15}$$

the term which contains both $\boldsymbol{\epsilon}_k$ and $\boldsymbol{\zeta}_j$ is omitted, and thus

$$\begin{aligned} \boldsymbol{Z}_{jk} &= \begin{bmatrix} \boldsymbol{D}^{\mathrm{T}}(\boldsymbol{T}_{op,k}, \boldsymbol{p}_{op,j})^{\odot} & \boldsymbol{D}^{\mathrm{T}}\boldsymbol{T}_{op,k}\boldsymbol{D} \end{bmatrix} \\ \boldsymbol{\delta}\boldsymbol{x}_{jk} &= \begin{bmatrix} \boldsymbol{\epsilon}_k^{\mathrm{T}} & \boldsymbol{\zeta}_j^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{16}$$

with definition of operator $^{\odot}$

$$\begin{bmatrix} \boldsymbol{\varepsilon}_{3\times 1} \\ \eta \end{bmatrix}_{4\times 1}^{\odot} = \begin{bmatrix} \eta\boldsymbol{1}_{3\times 3} & -\boldsymbol{\varepsilon}_{3\times 1}^{\wedge} \\ \boldsymbol{0}_{1\times 3} & 0 \end{bmatrix} \tag{17}$$

The linearizion of $\boldsymbol{\pi}(\boldsymbol{x})$ is

$$\begin{aligned} \boldsymbol{\pi}(\boldsymbol{x}_{jk}) =& \boldsymbol{s}(\boldsymbol{z}(\boldsymbol{x}_{jk})) \\ \approx& \boldsymbol{s}(\boldsymbol{z}(\boldsymbol{x}_{op,jk})) + \boldsymbol{S}_{jk}\boldsymbol{Z}_{jk}\boldsymbol{\delta}\boldsymbol{x}_{jk} \\ \approx& \boldsymbol{\pi}(\boldsymbol{x}_{op,jk}) + \boldsymbol{\Pi}_{jk}\boldsymbol{\delta}\boldsymbol{x}_{jk} \end{aligned} \tag{18}$$

where $\boldsymbol{S}_{jk}$ is the Jacobian of function $\boldsymbol{s}$. Thus the reprojection error is $\boldsymbol{e}_{y,jk}(\boldsymbol{x}) \approx \boldsymbol{e}_{y,jk}(\boldsymbol{x_{op}}) - \boldsymbol{\Pi}_{jk}\boldsymbol{\delta}\boldsymbol{x_{jk}}$.
The objective function can then be rearranged as:

$$J(\boldsymbol{x}) \approx J(\boldsymbol{x}_{op}) - \boldsymbol{b}^{\mathrm{T}}\delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\delta\boldsymbol{x} \tag{19}$$

where

$$\begin{aligned} \boldsymbol{b} &= \boldsymbol{H}^{\mathrm{T}}\boldsymbol{Q}^{-1}\boldsymbol{e}(\boldsymbol{x}_{op}) \\ \boldsymbol{A} &= \boldsymbol{H}^{\mathrm{T}}\boldsymbol{Q}^{-1}\boldsymbol{H} \end{aligned} \tag{20}$$

$$\begin{aligned} \boldsymbol{H} &= \left[\boldsymbol{H}_{10}^{\mathrm{T}}, ..., \boldsymbol{H}_{M0}^{\mathrm{T}}, \boldsymbol{H}_{11}^{\mathrm{T}}, ..., \boldsymbol{H}_{MK}^{\mathrm{T}}\right]^{\mathrm{T}} \\ \boldsymbol{Q} &= \mathrm{diag}\{\boldsymbol{Q}_{10}, ..., \boldsymbol{Q}_{M0}, \boldsymbol{Q}_{11}, ..., \boldsymbol{Q}_{MK}\} \\ \boldsymbol{e}(\boldsymbol{x}_{op}) &= \left[\boldsymbol{e}_{y,10}(\boldsymbol{x}_{op})^{\mathrm{T}}, ..., \boldsymbol{e}_{y,MK}(\boldsymbol{x}_{op})^{\mathrm{T}}\right]^{\mathrm{T}} \\ \boldsymbol{H}_{jk} &= \boldsymbol{\Pi}_{jk}\boldsymbol{P}_{jk} \end{aligned} \tag{21}$$

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

$$\begin{aligned} \frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{\delta x}^{\mathrm{T}}} &= -\boldsymbol{b} + \boldsymbol{A}\boldsymbol{\delta x} = 0 \\ \boldsymbol{A}\boldsymbol{\delta x}^{*} &= \boldsymbol{b} \end{aligned} \tag{22}$$

when optimal perturbation $\boldsymbol{\delta x}^{*}$ is obtained, the operating points are updated by:

$$\begin{aligned} \boldsymbol{T}_{op,k} &\leftarrow \exp(\boldsymbol{\epsilon}_{\boldsymbol{k}}^{*\wedge})\boldsymbol{T}_{op,k} \\ \boldsymbol{p}_{op,j} &\leftarrow \boldsymbol{p}_{op,j} + \boldsymbol{D}{\boldsymbol{\zeta}^{*}}_{j} \end{aligned} \tag{23}$$

The size of $\boldsymbol{A}$ is large, so the equation $\boldsymbol{A}\boldsymbol{\delta x}^{*} = \boldsymbol{b}$ is solved by exploring the sparsity pattern of $\boldsymbol{A}$ as stated in [25].

3.2. Sliding window filter setup

Set $t = 0, ..., T$ as the total time step for the overall observed trajectory. The window with size K slides along the time step t. The overall estimated states of the first window will be recorded. Only the last state will be recorded to achieve online performance for the rest of the windows. The initial state $\boldsymbol{T}_{b_0c}$ and $\boldsymbol{p}_{j0}$ for the first window is generated by Schmidt Orthogonalization in algorithm 1. Accordingly, initial state $\boldsymbol{T}_{b_0c}$ is calculated to be

Algorithm 1 Schmidt Orthogonalization

Input: measurement $\boldsymbol{y}_{j0} = [u_{j0}, v_{j0}, d_{j0}]^{\mathrm{T}}$
Output: $\boldsymbol{x}_b$,$\boldsymbol{y}_b$,$\boldsymbol{z}_b$ and $\boldsymbol{o}_b$

1: Calculate the 3D position of feature points in camera frame $\boldsymbol{p_{c0}}_{\boldsymbol{j}}$ by using Eq.(7);
2: Choose three points from step 1: $\boldsymbol{p}_{c1}$,$\boldsymbol{p}_{c2}$,$\boldsymbol{p}_{c3}$;
3: Set $\boldsymbol{p}_{c1}$ as the origin point $\boldsymbol{o}_b$, and compute $\boldsymbol{x}_b = \frac{\boldsymbol{p}_{c2}-\boldsymbol{p}_{c1}}{\|\boldsymbol{p}_{c2}-\boldsymbol{p}_{c1}\|_2}$ as x axis of object frame $\boldsymbol{B}$;
4: Set the vector $\boldsymbol{p}_y = \boldsymbol{p}_{c3} - \boldsymbol{p}_{c1}$ and calculate $\boldsymbol{y}_b$ by: $\bar{\boldsymbol{y}}_b = \boldsymbol{p}_y - \frac{\boldsymbol{x}_b^{\mathrm{T}}\boldsymbol{p}_y}{\boldsymbol{x}_b^{\mathrm{T}}\boldsymbol{x}_b}\boldsymbol{x}_b$ and $\boldsymbol{y}_b = \frac{\bar{\boldsymbol{y}}_b}{\|\bar{\boldsymbol{y}}_b\|}$;
5: Set $\boldsymbol{z}_b = \boldsymbol{x}_b \times \boldsymbol{y}_b$;

$$\begin{aligned} \boldsymbol{T}_{b_0c} &= \begin{bmatrix} \boldsymbol{R}_{b_0c} & \boldsymbol{r}_{b_0}^{cb_0} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \\ \boldsymbol{R}_{b_0c} &= \begin{bmatrix} \boldsymbol{x}_b & \boldsymbol{y}_b & \boldsymbol{z}_b \end{bmatrix}^{\mathrm{T}} \quad \boldsymbol{r}_{b_0}^{cb_0} = -\boldsymbol{R}_{b_0c}\boldsymbol{o}_b \end{aligned} \tag{24}$$

The initial state $\boldsymbol{p}_j$ is generated by

$$\boldsymbol{p}_{j0} = \boldsymbol{T}_{b_0c}\boldsymbol{p}_{cj} \tag{25}$$

For the rest of the windows, the initial state of window l is the first estimated state of the window $l-1$. The initial guess for time step k in sliding window l is generated by OPnP[20], as summarized in algorithm2.

Algorithm 2 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 $\check{\boldsymbol{T}}_{b_kc}$ by OPnP; when the measurement is insufficient $\check{\boldsymbol{T}}_{b_kc} = \boldsymbol{T}_{b_{(k-1)}c}$;
3: Use observation model and initial pose $\check{\boldsymbol{T}}_{b_kc}$ 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 $\check{\boldsymbol{p}}_{jk}$;

4. EXPERIMENTS

The performance of proposed method is studied on the simulated data. Two objects with different shapes are tested on two complicated trajectories. To test the efficacy of SWF, the estimated pose is compared with groundtruth; the error of SWF is compared with the one generated by OPnP to test the robustness given insufficient feature points.

Through all the experiment, the window size is set as 50. The parameters of RGB-D camera are: 15fps; depth resolution 640×480; RGB resolution 1600×1200; the covariance of measurement noise $\boldsymbol{Q}_{jk} = \mathrm{diag}\{\sigma_u^2, \sigma_v^2, \sigma_d^2\}$ are set as $\sigma_u = 5.249$ pixels, $\sigma_v = 3.937$ pixels and $\sigma_d = 0.01$m; the intrinsic calibration matrix with $f_u = f_v = 800$, $c_u = c_v = 0$.

4.1. Results from SWF

Two unknown objects are shown in Fig.2. The pose error is defined

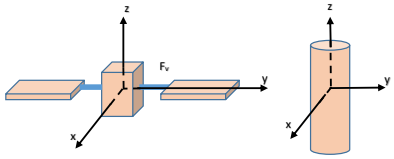


Fig. 2. Two Unknown Objects

as translational error and rotational error with estimated value x^{*}:

$$\begin{aligned} \delta\boldsymbol{r}_k &= \begin{bmatrix} \delta r_{x,k} & \delta r_{y,k} & \delta r_{z,k} \end{bmatrix}^{\mathrm{T}} := \bar{\boldsymbol{r}}_c^{b_kc^{*}} - \boldsymbol{r}_c^{b_kc} \\ \delta\boldsymbol{\theta}_k^{\times} &= \begin{bmatrix} \delta\theta_{x,k} \\ \delta\theta_{y,k} \\ \delta\theta_{z,k} \end{bmatrix}^{\times} := \boldsymbol{1} - \bar{\boldsymbol{R}}_{b_kc}^{*}\boldsymbol{R}_{b_kc}^{\mathrm{T}} \end{aligned} \tag{26}$$

Fig.3 and Fig.4 show the results of pose error, given the same measurement noise. The figures indicate that the error increase for the first several time steps and maintain at a constant level because the groundtruth is set as the first initial state for the simulation. Despite of the measurement noise, the estimated error comes from the first linearizations of the observation model.

Most of the estimation error lie within the 3σ envelopes, meaning that the estimator performs well, and the estimation results are reasonable. The corresponding 3σ envelopes and the estimation error from the two experiments are roughly at the same magnitude, which shows the consistency of the SWF-based framework on different objects and trajectories.

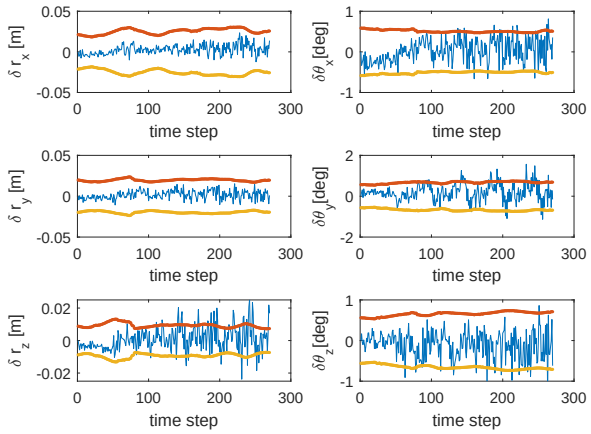


Fig. 3. Pose error for object 1. The blue solid line shows the estimation error. The yellow and orange lines show the corresponding 3σ envelopes.

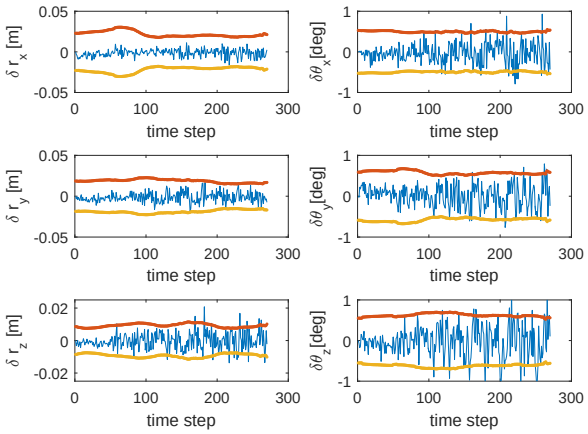
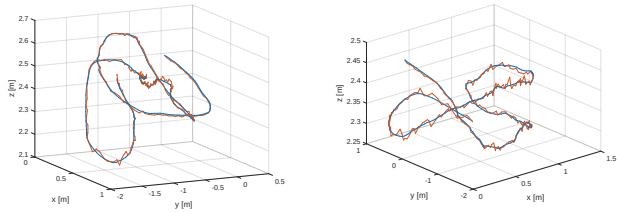


Fig. 4. Pose error for object 2. The blue solid line shows the estimation error. The yellow and orange lines show the corresponding 3σ envelopes.

Fig.5 shows the estimated trajectories. It shows that the SWF is able to estimate the complex trajectory especially for the mildly changing motion. Although there is jitter caused by the severely changed motion in the estimated trajectory, SWF can still track the object. An improvement that worth mentioning is that the proposed framework is free from the problem about closely singular coefficient matrix $\boldsymbol{A}$ in [4], and the rotation estimation is more robust and accurate than the result from [4].



(a) Trajectory of object1 (b) Trajectory of object2

Fig. 5. Trajectory Comparison. The blue lines are groundtruth; the orange lines are estimated trajectories.

4.2. Comparison with OPnP

To compare with OPnP, the following errors are defined as:

$$\boldsymbol{e}_{rk} = \|\delta \boldsymbol{r}_k\|_2, \quad \boldsymbol{e}_{\theta k} = \|\delta \boldsymbol{\theta}_k\|_2, \tag{27}$$

Fig.6 shows the performance comparison of OPnP and SWF. It shows that when the number of feature points is insufficient from time step 7 to time step 79, OPnP fails to generate the solution because OPnP requires at least 6 feature points to generate a unique solution. The proposed framework is more robust to the insufficient feature points because the estimated pose at time step $k-1$ is set as the initial guess at time step k when the number of feature points is less than 6. Meanwhile, the estimated structure which serves as the rigid body constraint is refined by SWF and thus the proposed framework is able to obtain the accurate pose given the inaccurate initial guess.

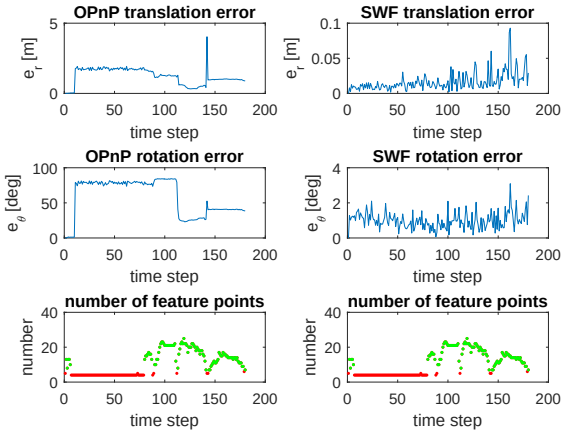


Fig. 6. Comparison of OPnP and SWF. The blue lines are $\boldsymbol{e}_r$ and $\boldsymbol{e}_\theta$ both for OPnP and proposed framework. The red dots are the number of feature points less than 6; the green are the number greater than 6.

5. CONCLUSIONS

This paper proposes a novel framework to estimate the structure and pose of an arbitrary unknown object on $SE(3)$ without any prior information about the object. The smoothing technique is adopted by using SWF and GN to explore the full current information. The GN method minimizes the reprojection error within a window of states to get rid of singularity of coefficient matrix and to enhance the estimation efficacy. The experiment shows the accuracy and robustness of the framework, given two objects with different shape and trajectories.

6. REFERENCES

[1] Hanme Kim, Stefan Leutenegger, and Andrew J Davison, "Real-time 3D reconstruction and 6-DoF tracking with an

event camera," in *European Conference on Computer Vision*. Springer, 2016, pp. 349–364.

[2] Nassir W Oumer and Giorgio Panin, "Tracking and pose estimation of non-cooperative satellite for on-orbit servicing," *i-SAIRAS 2012*, 2012.

[3] Gangqi Dong and Zheng Hong Zhu, "Vision-based Pose and Motion Estimation of Non-cooperative Target for Space Robotic Manipulators," in *AIAA SPACE 2014 Conference and Exposition*, 2014, p. 4263.

[4] Daniel R Canelhas, Todor Stoyanov, and Achim J Lilienthal, "SDF Tracker: A parallel algorithm for on-line pose estimation and scene reconstruction from depth images," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3671–3676.

[5] O Serdar Gedik and A Aydin Alatan, "RGBD data based pose estimation: Why sensor fusion?," in *Information Fusion (Fusion), 2015 18th International Conference on*. IEEE, 2015, pp. 2129–2136.

[6] Xue-Hai Gao, Bin Liang, Le Pan, Zhi-Heng Li, and Ying-Chun Zhang, "A monocular structured light vision method for pose determination of large non-cooperative satellites," *International Journal of Control, Automation and Systems*, vol. 14, no. 6, pp. 1535–1549, 2016.

[7] Nassir W Oumer and Giorgio Panin, "3D point tracking and pose estimation of a space object using stereo images," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 796–800.

[8] Zhengzhou Li, Fengzeng Ge, Wenhao Chen, Wanxing Shao, Bing Liu, and Bei Cheng, "Particle filter-based relative rolling estimation algorithm for non-cooperative infrared spacecraft," *Infrared Physics & Technology*, vol. 78, pp. 58–65, 2016.

[9] Roberto Opromolla, Giancarmine Fasano, Giancarlo Rufino, and Michele Grassi, "Uncooperative pose estimation with a LIDAR-based system," *Acta Astronautica*, vol. 110, pp. 287–297, 2015.

[10] D W Leng and W D Sun, "Contour-based iterative pose estimation of 3D rigid object," *IET computer vision*, vol. 5, no. 5, pp. 291–300, 2011.

[11] Ying He, Bin Liang, Xiaodong Du, Xueqian Wang, and Di Zhang, "Measurement of relative pose between two non-cooperative spacecrafts based on graph cut theory," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on*. IEEE, 2014, pp. 1900–1905.

[12] A Rubio, Michael Villamizar, Luis Ferraz, Adrián Penate-Sanchez, Arnau Ramisa, Edgar Simo-Serra, Alberto Sanfeliu, and Francesc Moreno-Noguer, "Efficient monocular pose estimation for complex 3D models," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1397–1402.

[13] Han Yu, Xiujie Zhang, Lingyu Liu, Shuo Wang, and Shenmin Song, "Relative dynamics estimation of non-cooperative spacecraft with unknown orbit elements and inertial tensor," *Chinese Journal of Aeronautics*, vol. 29, no. 2, pp. 479–491, 2016.

[14] Aitor Aldoma, Markus Vincze, Nico Blodow, David Gossow, Suat Gedikli, Radu Bogdan Rusu, and Gary Bradski, "CAD-model recognition and 6DOF pose estimation using 3D cues," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 585–592.

[15] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2155–2162.

[16] Daniel F DeMenthon and Larry S Davis, "Model-based object pose in 25 lines of code," in *European conference on computer vision*. Springer, 1992, pp. 335–343.

[17] C-P Lu, Gregory D Hager, and Eric Mjolsness, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, 2000.

[18] R Hartley, "[a]Minimizing Algebraic Error in Geometric Estimation Problem," in *[o] Proc. Sixth Int'l Conf. Computer Vision*, 1998, pp. 469–476.

[19] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, pp. 155, 2009.

[20] 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*, 2013, pp. 2344–2351.

[21] Luis Ferraz, Xavier Binefa, and Francesc Moreno-Noguer, "Very fast solution to the PnP problem with algebraic outlier rejection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 501–508.

[22] Shai Segal, Avishy Carmi, and Pini Gurfil, "Stereovision-based estimation of relative dynamics between noncooperative satellites: theory and experiments," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 2, pp. 568–584, 2014.

[23] Andrew Rhodes, Eric Kim, John A Christian, and Thomas Evans, "LIDAR-based Relative Navigation of Non-Cooperative Objects Using Point Cloud Descriptors," in *AIAA/AAS Astrodynamics Specialist Conference*, 2016, p. 5517.

[24] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza, "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry," *IEEE Transactions on Robotics*, 2016.

[25] Timothy D Barfoot, "STATE ESTIMATION FOR ROBOTICS," 2017.