Pose Estimation based on a Dual Quaternion Feedback Particle Filter

Wenjie Li[1], Wasif Naeem[2], Wenhao Ji[1], Jia Liu[1,*], Wei Hao[1], Lijun Chen[1,*]

***Abstract*— Fast and accurate pose estimation is essential for many robotic applications such as SLAM, manipulation, and 3D point registration. Existing solutions to this problem suffer from either high computation overhead due to the nonlinear features or accuracy loss due to linear approximation. In this paper, we propose a dual quaternion feedback particle filter (DQFPF) that can capture the nonlinear factors in the observation model and use the optimal control theory to estimate the pose. To avoid particle degeneracy caused by sequential importance sampling and resampling, we present a feedback particle update formula to speed up the optimization with fewer particles being sampled. Simulation results show that in known corresponding cases our approach can converge to the correct pose more efficiently than the state-of-the-art. A similar conclusion can also be drawn in real applications of unknown corresponding cases, i.e., point cloud stitching and visual odometry estimation.**

I. INTRODUCTION

Pose estimation (rotation and translation) between two consecutive sensor frames paves the way for many robotic applications, ranging from visual SLAM [1], manipulation [2], to 3D point registration [3] and augmented reality [4]. In recent years, probabilistic optimization approaches have attracted increasing attention due to their robustness in noisy measurements. A number of nonlinear probabilistic filters have been applied to the pose estimation because of two competitive advantages. First, they have the ability to capture nonlinear features during the implementation of the nonlinear observation model. Second, the nonlinear filters such as the particle filter can continuously update state variables utilizing the Monte Carlo techniques to generate an accurate state.

However, the use of nonlinear filters still suffers from two limitations. First, the pose representation generally lies on a non-Euclidean space $SE(3)$, which differs from the Euclidean space used in most nonlinear filters. Second, it needs to sample and update numerous particles during the implementation of several Monte Carlo methods such as the bootstrap particle filter [5] and the Rao-Blackwellized particle filter [6], which largely increases the execution time. Even worse, resampling is also required to avoid the particle degeneracy problem, which further lowers the time efficiency.

To tackle this problem, prior work chooses to estimate the pose by linearizing the nonlinear observation model. For instance, Srivatsan et al. [7] measure the observation by means of dual quaternions and then optimize the model under the framework of the linear Kalman filter. This linear approximation generally functions well but might give rise to inaccurate estimation or errors in some cases. Hence, this arises an interesting question: *can we design a time-efficient and accurate filter to estimate the pose online when considering nonlinear factors in the observation model?*

In this paper, we intend to answer this question and propose a dual quaternion feedback particle filter (DQFPF) that can capture the nonlinear factors in the observation model and use the optimal control theory to estimate the pose. To avoid particle degeneracy caused by sequential importance sampling and resampling, we present a feedback particle update formula to speed up the optimization with fewer particles being sampled, which makes DQFPF estimate the pose more accurately and time-efficiently. The main contributions of this work are three-fold.

- We derive the noisy observation model using dual quaternions by setting the expectation of real measurements as a non-zero value which is a more prevalent assumption in real scenarios.
- We present a robust dual quaternion feedback particle filter (DQFPF) for online pose estimation. A feedback particle update formula is adopted to avoid the sequential importance sampling and resampling steps. Moreover, the feedback mechanism ensures robustness in noisy models.
- We conduct extensive experiments on the simulated environments with known correspondence and real applications with unknown correspondence to demonstrate its better pose accuracy and time efficiency of the proposed method, compared to the state-of-the-art.

II. RELATED WORK

Pose estimation has been of interest for a long time in the robotic area. A lot of prior work has been presented which roughly falls into two categories: non-probabilistic optimization and probabilistic optimization approaches.

A. Non-probabilistic Optimization Approaches

Non-probabilistic optimization approaches mainly refer to the ICP method and its variants. A prominent approach for the ICP method is to leverage a robust function to reduce the effect of outliers [8]–[11] instead of explicitly pruning them. Aleksandr et al [12] then incorporated measurement uncertainties into a Generalized-ICP framework which further generalized the ICP-related methods. In light of the computational complexity of the original ICP method,

[1] The authors with the Department of Computer Science and Technology, Nanjing University, China. Email: {wenjielee,wenhaoji,hw}@smail.nju.edu.cn, {jialiu,chenlj}@nju.edu.cn.

[2] The author with the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, UK. Email: w.naeem@ee.qub.ac.uk.

[*] Corresponding author

S.Hong et al. [13] later proposed a fast ICP algorithm to estimate the pose, which is more computationally efficient. Besides, Bouaziz et al. [14] extended the ICP method by formulating the registration optimization using sparsity-inducing norms. Nevertheless, due to the consideration of all sensor measurements, the ICP method and its variants may have trouble in keeping a real-time performance in some robotic applications.

B. Probabilistic Optimization Approaches

Recently, probabilistic optimization approaches are being regarded as a promising research area because of their abilities in only adopting partial sensor measurements [15]–[17]. S.Hauberg et al. [16] proposed an unscented Kalman filter to improve the accuracy of the pose via the optimization of the nonlinear observation model using partial observation measurements. Likewise, K.Fathian et al. [18] presented an iterative extended Kalman filter to optimize the nonlinear model. Considering the representation of the pose, Aksel et al. [19] presented a dual quaternion particle filter for online pose estimation since dual quaternions can accurately represent both rotations and translations. Kaikai et al. [20] proposed to utilize a dual quaternion unscented Kalman filter combined with the particle filter to estimate the pose. However, nonlinear filters sometimes may have trouble in balancing the accuracy and time efficiency. In this case, to linearize the nonlinear observation model is an alternative pipeline. For instance, A.Srivatsan et al. [7], [21] developed a Bingham filter to estimate the pose by linearizing the observation model with dual quaternions.

Furthermore, several learning-based probabilistic optimization methods are also presented [2], [22]. Julian et al. [23] proposed to estimate the pose using a self-supervised learning technique. Gilitschenski et al. [24] and Peretroukhin et al. [25] proposed to estimate the pose combining CNNs with the statistical distributions. However, in many real-time robotic applications, we need to estimate the pose online with fewer computation resources being consumed. Moreover, massive training data are not always obtained in advance. Hence, non-learning methods still play a significant role in online pose estimation.

In this work, we present a dual quaternion feedback particle filter to the pose estimation. We (1) formulate the noisy non-linear observation model with dual quaternions alike to [7] but set the expectation of measurements as a non-zero value and then (2) present a robust dual quaternion feedback particle filter (DQFPF) for online pose estimation in which a feedback particle update formula is introduced to avoid the sequential importance sampling and resampling steps. Finally, we (3) conduct the proposed method on both simulations and real applications to demonstrate its accuracy and time efficiency.

III. Mathematic Model

A. Noise-free Model

The pose estimation can be mathematically represented with a dual quaternion form which is given in [21], that is, $\hat{\mathbf{m}} \odot \mathbf{Q}^{1*} = \mathbf{Q} \odot \hat{\mathbf{n}}$, where $\mathbf{Q} = \mathbf{q}_r + \varepsilon \mathbf{q}_d, \varepsilon^2 = 0$ indicates the dual quaternion which consists of two quaternions $\mathbf{q}_r = [s_{q_r}, \mathbf{v}_{q_r}] \in \mathbb{R}^4$ and $\mathbf{q}_d = [s_{q_d}, \mathbf{v}_{q_d}] \in \mathbb{R}^4$, the dual part $\mathbf{q}_d = \frac{\mathbf{q}_r \mathbf{q}_t}{2}$ is the composition of the rotation quaternion $\mathbf{q}_r$ and translation quaternion $\mathbf{q}_t \in \mathbb{R}^4$. Additionally, $\mathbf{Q}^{1*} = \mathbf{q}_r - \varepsilon \mathbf{q}_d$ represents the conjugate of the dual quaternion.

Let $\hat{\mathbf{m}} = \mathbf{1} + \varepsilon \mathbf{m}$ be a 3D point[1] denoted by the dual quaternion. Consider two consecutive measurement points $\mathbf{m}_1, \mathbf{m}_2$ and their corresponding points $\mathbf{n}_1, \mathbf{n}_2$. We have

$$\hat{\mathbf{m}}_1 \odot \mathbf{Q}^{1*} = \mathbf{Q} \odot \hat{\mathbf{n}}_1, \tag{1a}$$

$$\hat{\mathbf{m}}_2 \odot \mathbf{Q}^{1*} = \mathbf{Q} \odot \hat{\mathbf{n}}_2. \tag{1b}$$

Then the difference between $(1a)$ and $(1b)$ is $(\mathbf{m}_1 - \mathbf{m}_2)\,\mathbf{q}_r - \mathbf{q}_r\,(\mathbf{n}_1 - \mathbf{n}_2) = \mathbf{0}$.

Next, we expand it in accordance with the multiplication of quaternions,

$$\begin{aligned} &\begin{bmatrix} 0 & -(\mathbf{v}_{m1} - \mathbf{v}_{m2})^T \\ (\mathbf{v}_{m1} - \mathbf{v}_{m2}) & (\mathbf{v}_{m1} - \mathbf{v}_{m2})^\wedge \end{bmatrix} \mathbf{q}_r \\ &- \begin{bmatrix} 0 & -(\mathbf{v}_{n1} - \mathbf{v}_{n2})^T \\ (\mathbf{v}_{n1} - \mathbf{v}_{n2}) & -(\mathbf{v}_{n1} - \mathbf{v}_{n2})^\wedge \end{bmatrix} \mathbf{q}_r = \mathbf{0}, \\ &\Rightarrow \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{bmatrix} \mathbf{q}_r = \mathbf{0}, \\ &\Rightarrow \mathbf{M}_{4\times4}\mathbf{q}_r = \mathbf{0}, \end{aligned} \tag{2}$$

where $\mathbf{M}_{4\times4}$ denotes the observation matrix[2],

$$\begin{aligned} \mathbf{M}_{11} &= 0_{1\times1}, \\ \mathbf{M}_{12} &= -\left(\mathbf{v}_{m1} - \mathbf{v}_{m2} - \mathbf{v}_{n1} + \mathbf{v}_{n2}\right)^T, \\ \mathbf{M}_{21} &= \left(\mathbf{v}_{m1} - \mathbf{v}_{m2} - \mathbf{v}_{n1} + \mathbf{v}_{n2}\right), \\ \mathbf{M}_{22} &= \left(\mathbf{v}_{m1} - \mathbf{v}_{m2} + \mathbf{v}_{n1} - \mathbf{v}_{n2}\right)^\wedge. \end{aligned}$$

Meanwhile, add $(1a)$ and $(1b)$. We have

$$\begin{aligned} &(\mathbf{m}_1 + \mathbf{m}_2)\,\mathbf{q}_r - \mathbf{q}_r\,(\mathbf{n}_1 + \mathbf{n}_2) = 2\mathbf{q}_t\mathbf{q}_r, \\ &\Rightarrow \mathbf{q}_t = \frac{1}{2}(\mathbf{m}_1 + \mathbf{m}_2) - \frac{1}{2}\mathbf{q}_r\,(\mathbf{n}_1 + \mathbf{n}_2)\,\mathbf{q}_r^*, \end{aligned} \tag{3}$$

where $\mathbf{q}_r^*$ refers to the conjugate of $\mathbf{q}_r$.

Hence, the observation model can be drawn with two separated but interrelated equations shown in (2) and (3).

B. Noisy Observation Model

The pose estimation problem can be visualized in Fig. 1. Yet the observation equation (2) doesn't strictly hold in real applications due to uncertainties. Consequently, a noisy observation model is necessary to be derived in this work.

Here we rewrite (2) with a more general form[3],

$$\mathbf{u} = \mathbf{M}\mathbf{q}_r = \mathbf{N}\left(\mathbf{q}_r\right)\boldsymbol{\sigma}_{true}, \mathbf{q}_r = [s_r, \mathbf{v}_r]^T, \tag{4}$$

[1]The 3D point $\mathbf{v}_m$ is represented by the quaternion $\mathbf{m} = [0, \mathbf{v}_m]$.

[2]$(\cdot)^\wedge$ denotes a skew symmetric matrix.

[3]Note: $\mathbf{u} = 0$ holds in a noise-free environment.

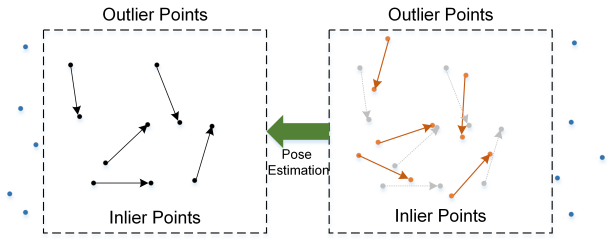

Fig. 1. A simple illustration of pose estimation, where the pose is estimated by aligning the right pairwise orange points to the left pairwise black points. In unknown corresponding cases, the outlier blue points will be filtered out which can be seen in Algorithm 3.

where $\mathbf{M} \in \mathbb{R}^{4\times4}$, $\mathbf{q}_r \in \mathbb{R}^{4\times1}$, $\mathbf{N}(\mathbf{q}_r) = [\mathbf{N}_1, \mathbf{N}_2, \mathbf{N}_3, \mathbf{N}_4]$, and $\boldsymbol{\sigma}_{true} = \left[\mathbf{v}_{m1}^T, \mathbf{v}_{m2}^T, \mathbf{v}_{n1}^T, \mathbf{v}_{n2}^T\right]^T$, where

$$\mathbf{N}_1 = \begin{bmatrix} -\mathbf{v}_r^T \\ -\mathbf{v}_r^\wedge + s_r\mathbf{I}_3 \end{bmatrix}, \mathbf{N}_2 = -\mathbf{N}_1,$$
$$\mathbf{N}_3 = \begin{bmatrix} \mathbf{v}_r^T \\ -\mathbf{v}_r^\wedge - s_r\mathbf{I}_3 \end{bmatrix}, \mathbf{N}_4 = -\mathbf{N}_3.$$

Moreover, the variable $\boldsymbol{\sigma}_{true}$ refers to the true sensor measurement in a noise-free environment. Then we define the variable $\boldsymbol{\sigma}$ as the sensor measurement in a real environment with noise $\delta\boldsymbol{\sigma}$, where $\delta\boldsymbol{\sigma} = \boldsymbol{\sigma} - \boldsymbol{\sigma}_{true}$, satisfying $\delta\boldsymbol{\sigma} \sim N(\mathbf{0}, \boldsymbol{\Omega}^{\boldsymbol{\sigma}})$. Hence we have

$$\mathbf{u} = \mathbf{N}(\mathbf{q}_r)\,\boldsymbol{\sigma} = \mathbf{N}(\mathbf{q}_r)\,(\delta\boldsymbol{\sigma} + \boldsymbol{\sigma}_{true}) = \mathbf{N}(\mathbf{q}_r)\,\delta\boldsymbol{\sigma}, \quad (5)$$

where $\mathbf{u} \in \mathbb{R}^4 \sim N(\mathbf{0}, \boldsymbol{\Omega}^{\mathbf{u}})$.

According to the stochastic theory proposed in [26], $\mathbf{q}_r$, $\delta\boldsymbol{\sigma}$, $\mathbf{u}$ are mutual independent, with the mean $\mathbf{E}(\mathbf{q}_r)$, $\mathbf{0}$, $\mathbf{0}$ and covariance matrix $\boldsymbol{\Omega}^{\mathbf{q}_r}$, $\boldsymbol{\Omega}^{\boldsymbol{\sigma}}$, $\boldsymbol{\Omega}^{\mathbf{u}}$ respectively. Then the covariance matrix $\boldsymbol{\Omega}^{\mathbf{u}}$ is given by

$$\boldsymbol{\Omega}^{\mathbf{u}} = \mathbf{N}(\mathbf{q}_r)\,\boldsymbol{\Omega}^{\sigma}\mathbf{N}^T(\mathbf{q}_r) + \boldsymbol{\Delta}\,(\boldsymbol{\Omega}^{\sigma} \otimes \boldsymbol{\Omega}^{\mathbf{q}_r})\,\boldsymbol{\Delta}^T, \quad (6)$$

where $\otimes$ refers to Kronecker product and $\boldsymbol{\Delta}$ is defined as follows $\boldsymbol{\Delta} \triangleq [\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2, ..., \boldsymbol{\Delta}_n]$. Each sub-matrix $\boldsymbol{\Delta}_i, i = 1, 2, ..., n$ is obtained according to the identity $\boldsymbol{\Delta}_i = \mathbf{N}(\mathbf{q}_r)\,\mathbf{e}_i$, where $\mathbf{e}_i$ is the unit vector with 1 at position i and 0 otherwise.

Unlike existing work [21] that takes the noisy observation $\boldsymbol{\sigma}$ as a zero mean value. Instead, we leverage $\delta\boldsymbol{\sigma}$ to estimate the real rotation quaternion $\mathbf{q}_r$. Likewise, the translation quaternion $\mathbf{q}_t$ is obtained using the stochastic theory and (3) in noisy scenarios.

IV. Dual Quaternion Feedback Particle Filter

The feedback particle filter presented in [27] is a new formulation of the particle filter, which is based on the optimal control theory and the mean-field game theory. In this paper, we propose a dual quaternion feedback particle filter to estimate the pose online.

We first formulate the pose estimation as a discrete control problem which is shown as follows,

$$\mathbf{q}^k = \mathbf{h}\mathbf{q}^{k-1} + \mathbf{B}^{k-1}, \quad (7a)$$
$$\mathbf{u}^k = \underbrace{\mathbf{M}\mathbf{q}^k}_{\mathbf{g}(\mathbf{q}^k)} + \mathbf{W}^k, \quad (7b)$$

where the quaternion $\mathbf{q}^k \in \mathbb{R}^{4\times1}$ is the state vector at the time t_k, the $\mathbf{h} \in \mathbb{R}^{4\times4}$ and $\mathbf{M} \in \mathbb{R}^{4\times4}$ are the state transition matrix and the observation matrix respectively, and $\mathbf{B}^{k-1}, \mathbf{W}^k$ are process and observation noise which are mutually independent.

Alike to normal particle filters, the objective of the feedback particle filter is to calculate or approximate the posterior distribution of $\mathbf{q}^k$ given the observation history $\mathbf{u}^{1:k}$, which is denoted as $P(\mathbf{q}^k|\mathbf{u}^{1:k})$. It is critical to build N stochastic processes $\{\mathbf{q}_i^k : 1 \le i \le N\}$, where $\mathbf{q}_i^k$ indicates the ith particle at the time t_k. And for each time interval, the distribution of the state is approximated by the mean of particle population, which is shown as follows,

$$P(\mathbf{q}^k|\mathbf{u}^k) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\{\mathbf{q}_i^k \in \mathbf{O}\}, \quad (8)$$

where $\mathbf{O} \subset \mathbb{R}$ is the observation set.

In order to update particles at each time step, a universal technique called sequential importance sampling (SIS) is necessary in which particles are sampled according to their importance weights and the accuracy of the state variable is improved by choosing a proper sampling method. Moreover, resampling is also required to avoid particle degeneracy problem, which is time-consuming sometimes as the number of particles increases yet.

Here we abandon this mechanism and update the ith particle from the perspective of optimal control theory in which the update formulation is given by

$$\mathbf{q}_i^{t+\Delta t} = \mathbf{h}\mathbf{q}_i^t + \mathbf{B}_i^t\Delta t + \mathbf{U}_i^t, \quad (9)$$

where $\mathbf{U}_i^t$ is the designed control input on the ith particle at the time t. Generally, the control input $\mathbf{U}_i^t$ is denoted as $\mathbf{U}_i^t = \mathbf{K}_i^t\mathbb{I}_i^t + \boldsymbol{\Omega}_i^t$, where $\mathbf{K}_i^t \in \mathbb{R}^{m\times n}$ is the feedback gain matrix, $\boldsymbol{\Omega}_i^t$ refers to the Wong-Zakai correction term [28], and $\mathbb{I}_i^t$ is the innovation error, which is represented as

$$\mathbb{I}_i^t = \mathbf{y}^t - \frac{1}{2}(\mathbf{g}(\mathbf{q}_i^t) + \bar{\mathbf{g}}), \quad (10)$$

where $\bar{\mathbf{g}} \approx \frac{1}{N}\sum_{i=1}^{N}\mathbf{g}(\mathbf{q}_i^t)$.

However, the main difficulty of the feedback particle filter is to obtain the feedback gain $\mathbf{K}_i^t$, which is solved by computing a weighted Poisson equation:

$$\nabla(p\nabla\phi_j) = -(h_j - \bar{h}_j)p/Q_{jj}, \quad (11)$$

where Q_{jj} is the covariance of the jth particle in $\mathbf{u}^k$, the function ϕ_j is the solution of (11). Then the feedback gain matrix is given by $\mathbf{K}_{lj}^t = \frac{\alpha\phi_j}{\alpha\mathbf{q}_l}$.

Here we choose to generate the feedback gain $\mathbf{K}$ utilizing the constant gain approximation technique since it can get a competitive performance with fewer parameters being tuned [29], which can be found in Algorithm 2. Furthermore, we focus on the static pose estimation. Hence, the state transition matrix $\mathbf{h}$ is the identity matrix.

The proposed dual quaternion feedback particle filter can be seen in Algorithm 1. First, the initial rotation quaternion is obtained from the given dual quaternion $\mathbf{Q}_{init}$ in Line

1. Then a pseudo-time interval Δt is introduced so that a simulated-based update can be performed to correct the predicted particles' state in Line 2. Next, we sample N particles $\mathbf{q}^0$ from the initial distribution $P(\mathbf{q}, 0)$.

Algorithm 1: Dual Quaternion Feedback Particle Filter

Input : Point cloud A and B, initial dual quaternion $\mathbf{Q}_{init}$, time T, the number of particles N
Output: Estimated dual quaternion $\mathbf{Q}_{est}$

1 Generate $\mathbf{q}_{init}$ from $\mathbf{Q}_{init}$;
2 Pseudotime interval $\Delta t \leftarrow 0.1$;
3 $t \leftarrow 0$;
4 Sample N particles $\mathbf{q}^0 \leftarrow \{\mathbf{q}_1^0, \mathbf{q}_2^0, \mathbf{q}_3^0, ..., \mathbf{q}_N^0\}$;
5 Normalize each particle Norm($\mathbf{q}_i^0$);
6 $\mathbf{q}_{updated} \leftarrow []$;
7 **while** $t \leq T$ **do**
8 $\quad g(\mathbf{q}_i^t) \leftarrow$ Eq. (7b);
9 $\quad \bar{g}(\mathbf{q}^t) \leftarrow \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{q}_i^t)$;
10 $\quad \mathbf{K}^t \leftarrow$ Algorithm 2;
11 $\quad$ **for** *i=1 to N* **do**
12 $\quad\quad$ Generate a sample, ΔV, from $N(\mathbf{0}_{1\times 4}, \Delta t \mathbf{I}_{4\times 4})$;
13 $\quad\quad$ Calculate $\Delta \mathbb{I}_i^t \leftarrow -\frac{1}{2}(g(\mathbf{q}_i^t + \bar{g}(\mathbf{q}^t)))\Delta t$;
14 $\quad\quad \mathbf{q}_i^{t+\Delta t} \leftarrow \mathbf{q}_i^t + \mathbf{K}^t \Delta \mathbb{I}_i^t + 0.01\Delta V\sqrt{\Delta t}\mathbf{I}_{4\times 4}$;
15 $\quad\quad \mathbf{q}_i^{t+\Delta t} \leftarrow \frac{\mathbf{q}_i^{t+\Delta t}}{\|\mathbf{q}_i^{t+\Delta t}\|}$
16 $\quad$ **end**
17 $\quad \mathbf{q}_{update} \leftarrow \mathbf{q}^t$;
18 $\quad t \leftarrow t + \Delta t$;
19 **end**
20 $\mathbf{q}_{est} \leftarrow$Norm($\frac{1}{N}\sum \mathbf{q}_{update}^T$);
21 $\mathbf{t}_{est} \leftarrow$ Eq. (3);
22 Generate estimated $\mathbf{Q}_{est}$;

We update sampled particles in Line 7-19. First, the noisy observation $g(\mathbf{q}_i^t)$ is generated in Line 8 according to (7b), and the mean of observation values of all particles is also necessary for the computation of $\mathbb{I}_i^t$ in Line 13. Subsequently, we update each particle in Line 14 that is also presented in (9). Finally, the estimated rotation quaternion can be derived by averaging estimated particles, and the translation quaternion can also be obtained utilizing (3).

Algorithm 2: Constant Gain Algorithm

Input : Quaternion particles $\mathbf{q} = \{\mathbf{q}_i^t\}_{i=1}^N$
Output: Constant gain $\mathbf{K}(\mathbf{q}_i^t, t)$

1 $N \leftarrow len(q)$;
2 $g(\mathbf{q}_i^t) \leftarrow$ Eq. (7b);
3 $\bar{g}(\mathbf{q}^t) \leftarrow \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{q}_i^t)$;
4 $\mathbf{K}(\mathbf{q}_i^t, t) \leftarrow \frac{1}{N}\sum_{j=1}^{N} \mathbf{q}_i^t(g(\mathbf{q}_i^t) - \bar{g}(\mathbf{q}^t))$;

However, it is common that the corresponding points are unknown in two consecutive point clouds A and B. Here we also give our strategy shown in Algorithm 3. First, we downsample points from the given dense point cloud $\mathcal{A}$ and $\mathcal{B}$ in Line 1. We split the downsampled point cloud B into n patches which can lead to a faster convergence in Line 2. Then each patch is matched to the part of the A using the KD-tree [30] in Line 5-7. Next, we filter out outlier points using the given inlierRatio λ in Line 8-10. Finally, we give a decision on if the average distance is less than distance threshold γ. If not, Algorithm 1 will be implemented to generate a new dual quaternion and begin the next iteration.

Algorithm 3: Dual Quaternion Feedback Particle Filter with unknown correspondence

Input: Fixed point cloud $\mathcal{A}$, transformed point cloud $\mathcal{B}$, initial dual quaternion $\mathbf{Q}_{init}$, time T, the number of particles N, inlierRatio λ, maxIteration $iter$, patch size s, distance threshold γ
Output: Estimated dual quaternion $\mathbf{Q}_{est}$

1 A,$B \leftarrow$ downsample($\mathcal{A}$, $\mathcal{B}$);
2 $n \leftarrow \lfloor len(B)/s \rfloor$;
3 $\mathbf{Q}_{est} \leftarrow \mathbf{Q}_{init}$;
4 **for** *i=1 to iter* **do**
5 $\quad$ Generate n patches P_B^n, kdt $\leftarrow$KDTree(A);
6 $\quad$ For each patch, transform the patch area using (1a), $TP_A \leftarrow$ transform($P_B, \mathbf{Q}_{est}$);
7 $\quad$ Find closet points and closet distances, $TP_{As}, dist_s \leftarrow$ kdt.query(TP_A);
8 $\quad$ Calculate the number of inlier points, $In_P \leftarrow \lfloor len(TP_A) * \lambda \rfloor$;
9 $\quad TP_m, A_m \leftarrow$ FindInlier(TP_{As}, A);
10 $\quad Avg_d$ = Avg($dist_s$,In_P);
11 $\quad$ **if** $Avg_d \geq \gamma$ **then**
12 $\quad\quad \mathbf{Q}_{est} \leftarrow$ Algorithm1($TP_m, A_m, \mathbf{Q}_{est}, T, N$);
13 $\quad$ **end**
14 $\quad$ **else**
15 $\quad\quad$ break;
16 $\quad$ **end**
17 **end**

V. EXPERIMENTS

In this section, we apply our approach on point registration scenarios on both known and unknown corresponding cases. The performance is studied as follows, where the Kalman filter-based method (DQKF) [21], Bingham filter-based method (BMF) [7], and a modified particle filter-based method (DQPF) [19] are taken as the baseline for comparison[4].

A. Known Correspondence Case

We first randomly sample 952 points from the Stanford bunny dataset. We rotate it along x-axis with an rotation angle $\alpha = \pi/3$, i.e., rotation quaternion $\mathbf{q}_r =$

[4]All experiments that are coded by Python in this work are carried out on the Linux system with 8G RAM and Intel i7 processor.

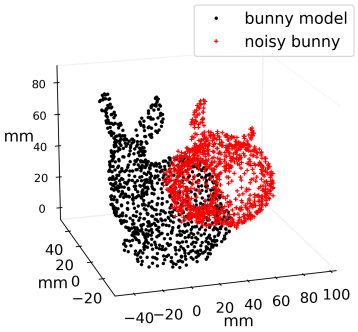

Fig. 2. The Stanford Bunny (black) and its noisy transformed form (red).

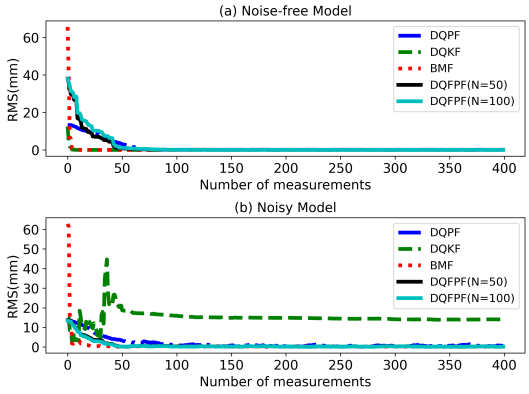

Fig. 3. (a) The RMS values in a noise-free Bunny model. (b) The RMS values in a noisy Bunny model.

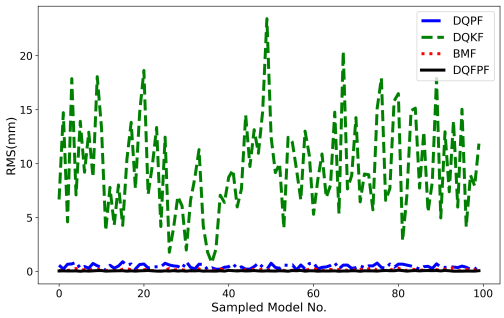

Fig. 4. The RMS values in different methods in 100 random noisy models.

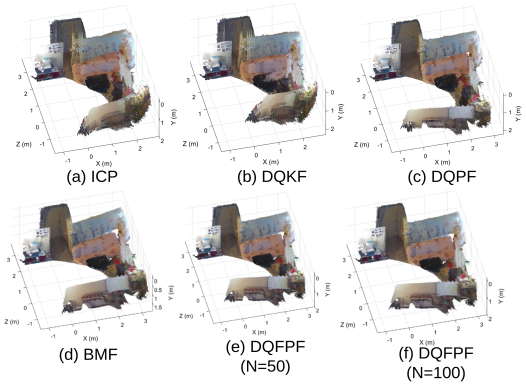

Fig. 5. The results of 3D construction using different methods. (a) The ICP method takes about 18.41s to finish the construction, and (b) the DQKF method takes about 11.52s, (c) the DQPF ($N = 500$) takes about 14.8s, (d) the BMF takes about 9.3s, and (e) the DQFPF ($N = 50$) takes about 6.9s, (f) the DQFPF ($N = 100$) takes about 9.9s.

$[0.866, 0.500, 0.000, 0.000]$. We translate the model along vector $\mathbf{t} = [50, 42, 20]$, i.e., $\mathbf{q}_t = [0, 50, 42, 20]$. Then we give the initial quaternion $\mathbf{q}_{r0} = [0.866, 0.452, 0.151, 0.151]$ and assume $\mathbf{q}_r$'s covariance $\Omega^{\mathbf{q}^r} = \mathbf{I}_{4\times4}$, the noise $\delta\sigma$'s covariance $\Omega^{\sigma} = \mathbf{I}_{12\times12}$. The initial particles are sampled from the Gaussian distribution $N(\mathbf{q}_r, 10^{-4}\mathbf{I}_{4\times4})$. Meanwhile, we set the time $T = 1$. We also set different number of particles $N = 50, 100$ to show the accuracy and runtime of our method.

We first run the proposed method on the noise-free bunny model. The results can be seen in Fig. 3(a), where the final errors are measured with the root mean square (RMS). We find that the RMS values of implemented approaches are able to converge to zero after limited measurements. Furthermore, from Table I, our method has less time consumed of 4.54s when $N = 50$, and 6.08s when $N = 100$ in which we take all measurement points into consideration.

Subsequently, we add the noise on the transformed Bunny depicted in Fig. 2 where the noise satisfies Gaussian distribution $N(\mathbf{0}, 9\mathbf{I}_{3\times3})$. Simulation results in Fig. 3(b) and Table I demonstrate that our method can get a more precise result which takes 5.2s with an RMS error of 0.143mm when $N = 50$, and 6.10s with an RMS error of 0.131mm when $N = 100$ in comparison with other methods.

To further demonstrate the performance of our method, we also randomly generate 100 transformed noisy bunny models by uniformly drawing noise covariance values from $[0, 1]$. The final RMS values are shown in Fig. 4. Obviously, our method ($N = 50$) represented by the black line has stronger adaptability in different noisy environments with the least RMS value when compared to the state-of-the-art.

B. Unknown Correspondence Case

Generally, unknown corresponding cases are more common in real scenarios. In this subsection, we perform the proposed algorithm on two real applications, i.e., point cloud stitching and visual odometry estimation and give a comparison with the state-of-the-art.

1) Point Cloud Stitching: We rebuild the 3D scene using 44 consecutive RGBD images obtained from the Kinect by estimating the relative pose between pairwise images using Algorithm 3, where the patch size $s = 150$, inlier ratio $\lambda = 0.5$, and distance threshold $\gamma = 0.4$.

From Fig. 5 and Fig. 6(a), the average RMS value of the proposed method is 0.022m when $N = 50$, and 0.020m when $N = 100$, which is lower than the BMF method of 0.03m and also performs better than the DQKF, the DQPF and the ICP methods. Additionally, our approach ($N = 50$) takes about 6.62s to construct the point cloud model (0.15s for each estimation in pairwise consecutive frames), which is faster than the DQKF which takes about 11.52s (0.26s for each estimation), the DQPF ($N = 50$)

TABLE I

THE ESTIMATED POSE BOTH IN NOISE-FREE BUNNY AND NOISY BUNNY MODEL

Model	Method	q_0	q_1	q_2	q_3	t_x	t_y	t_z	RMS(mm)	Time(s)
Noise-free Bunny	Actual	0.866	0.500	0.000	0.000	50.000	42.000	20.000	–	–
	DQPF	0.866	0.500	0.000	0.000	50.001	41.998	20.000	0.011	7.75
	DQKF	0.866	0.500	0.000	0.000	50.000	42.000	20.000	0.000	6.10
	BMF	0.866	0.500	0.000	0.000	50.000	42.000	20.000	0.000	6.18
	DQFPF(N=50)	0.866	0.500	0.000	0.000	50.001	42.000	20.000	0.001	4.54
	DQFPF(N=100)	0.866	0.500	0.000	0.000	50.000	42.000	20.001	0.001	6.08
Noisy Bunny	DQPF	0.864	0.502	-0.001	0.007	49.701	42.149	20.264	0.351	8.36
	DQKF	0.715	0.704	0.013	-0.015	50.293	43.782	36.759	14.077	6.04
	BMF	0.865	0.501	0.000	0.002	49.872	42.048	20.088	0.170	6.03
	DQFPF(N=50)	0.865	0.501	0.000	0.001	49.953	42.009	20.032	0.143	5.20
	DQFPF(N=100)	0.865	0.502	0.000	0.000	49.980	42.000	20.020	0.131	6.10

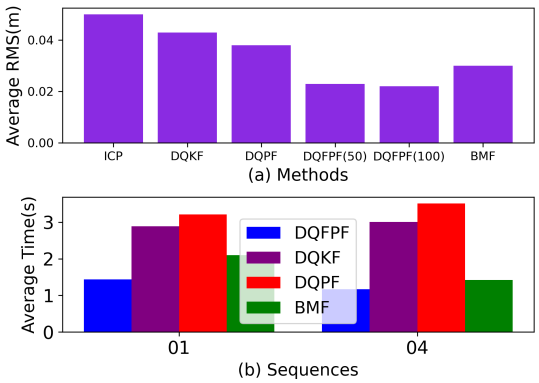

Fig. 6. (a) The average RMS values of different methods in point cloud stitching. (b) The average time of each estimation on pairwise point cloud data in sequence 01 and 04 in the KITTI. (Best view electronically.)

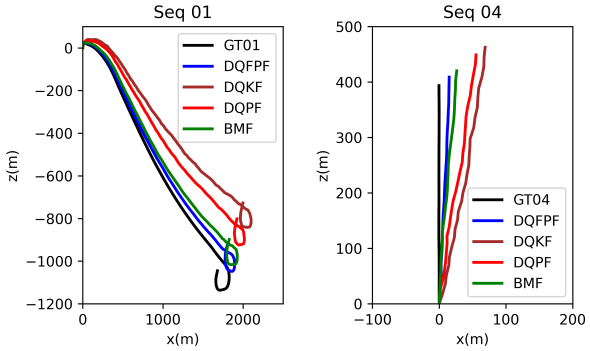

Fig. 7. The estimated trajectories in (a) sequence 01 and (b) sequence 04, where the proposed method denoted by blue line comes closet to the ground truth indicated by the black line. (Best view electronically.)

that takes 14.8s (0.3s for each estimation), and the BMF method that takes about 9.3s (0.21s for each estimation). Note that the ICP method is a common approach which takes all measurements into a global optimization framework. However, it is time-consuming that takes about 18.41s (0.41s for each estimation).

2) Pose Estimation of Visual Odometry: We apply the proposed method ($N = 50$) on the LiDAR data to estimate the relative pose of the visual odometry using a similar parameter configuration. We select sequences 01 and 04 in the KITTI as reference trajectories. Note that we only focus on the static pose estimation between two consecutive point clouds where the global optimization and loop closure are missing in this paper. As a result, the cumulative errors of estimated trajectories are doomed to be bigger with the increase of the length of trajectories. On the other hand, the cumulative errors will be smaller when estimated trajectories come closer to the ground truth which is our criterion of measuring different methods.

The final estimated trajectories are shown in Fig. 7. Our method denoted by the blue line comes closet to the ground truth on both sequences 01 and 04 with a relatively smaller cumulative error compared to other approaches, while the DQKF performs worst. Furthermore, from Fig. 6(b), we also find that the average estimated time of the proposed method, which is denoted by the blue bar, on pairwise point cloud data is about 1.41s in the sequence 01 and 1.17s in the sequence 04, while the DQPF method indicated by the red bar has a maximum time cost on both sequences.

VI. CONCLUSIONS

We present a dual quaternion feedback particle filter for online pose estimation. We first derive the noisy observation model using dual quaternions by setting the expectation of real measurements as a non-zero value. Then we present a robust dual quaternion feedback particle filter based on the acquired model which takes the optimal control as the vehicle. Moreover, a feedback particle update formula is used to avoid the sequential importance sampling and resampling so that it can run faster with fewer particles being sampled. Finally, we conduct simulations on the Bunny dataset with known correspondence showing that our method is robust to the noisy model which can converge to the correct pose after limited iterations in a few seconds. We also evaluate the proposed method on two real-world applications, i.e., point cloud stitching and visual odometry estimation, with unknown correspondence to further demonstrate its better accuracy and time efficiency compared to the state-of-the-art.

ACKNOWLEDGMENT

This research is financially supported by the National Natural Science Foundation of China under Grant 62072231 and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

REFERENCES

[1] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

[2] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1328–1342, 2021.

[3] G. D. Pais, S. Ramalingam, V. M. Govindu, J. C. Nascimento, R. Chellappa, and P. Miraldo, "3dregnet: A deep neural network for 3d point registration," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7191–7201.

[4] Z. Makhataeva and H. A. Varol, "Augmented reality for robotics: A review," *Robotics*, vol. 9, no. 2, p. 21, 2020.

[5] J. Candy, "Bootstrap particle filtering," *IEEE Signal Processing Magazine*, vol. 24, pp. 73–85, 2007.

[6] K. Murphy and S. Russell, "Rao-blackwellised particle filtering for dynamic bayesian networks," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 499–515.

[7] R. A. Srivatsan, M. Xu, N. Zevallos, and H. Choset, "Probabilistic pose estimation using a bingham distribution-based linear filter," *The International Journal of Robotics Research*, vol. 37, pp. 1610 – 1631, 2018.

[8] T. Masuda and N. Yokoya, "A robust method for registration and segmentation of multiple range images," in *Proceedings of 1994 IEEE 2nd CAD-Based Vision Workshop*, 1994, pp. 106–113.

[9] E. Trucco, A. Fusiello, and V. Roberto, "Robust motion and correspondence of noisy 3-d point sets with missing data," *Pattern recognition letters*, vol. 20, no. 9, pp. 889–898, 1999.

[10] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International journal of computer vision*, vol. 13, no. 2, pp. 119–152, 1994.

[11] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-icp: A globally optimal solution to 3d icp point-set registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2241–2254, 2016.

[12] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[13] S. Hong and C. Ye, "A pose graph based visual slam algorithm for robot pose estimation," in *2014 World Automation Congress*, 2014, pp. 917–922.

[14] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," in *Computer graphics forum*, vol. 32, no. 5. Wiley Online Library, 2013, pp. 113–123.

[15] F. Faion, P. Ruoff, A. Zea, and U. D. Hanebeck, "Recursive bayesian calibration of depth sensors with non-overlapping views," in *2012 15th International Conference on Information Fusion*, 2012, pp. 757–762.

[16] S. Hauberg, F. Lauze, and K. S. Pedersen, "Unscented kalman filtering on riemannian manifolds," *Journal of mathematical imaging and vision*, vol. 46, no. 1, pp. 103–120, 2013.

[17] M. H. Moghari and P. Abolmaesumi, "Point-based rigid-body registration using an unscented kalman filter," *IEEE Transactions on Medical Imaging*, vol. 26, no. 12, pp. 1708–1728, 2007.

[18] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans, "Quaternion based camera pose estimation from matched feature points," *arXiv preprint arXiv:1704.02672*, 2017.

[19] A. Sveier and O. Egeland, "Dual quaternion particle filtering for pose estimation," *IEEE Transactions on Control Systems Technology*, vol. 29, pp. 2012–2025, 2021.

[20] K. Li, F. Pfaff, and U. Hanebeck, "Unscented dual quaternion particle filter for se(3) estimation," *IEEE Control Systems Letters*, vol. 5, pp. 647–652, 2021.

[21] R. A. Srivatsan, G. T. Rosen, D. F. N. Mohamed, and H. Choset, "Estimating se (3) elements using a dual quaternion based linear kalman filter." in *Robotics: Science and systems*, 2016.

[22] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5738–5746.

[23] J. Nubert, S. Khattak, and M. Hutter, "Self-supervised learning of lidar odometry for robotic applications," in *2021 IEEE International Conference on Robotics and Automation*, 2021, pp. 9601–9607.

[24] I. Gilitschenski, R. Sahoo, W. Schwarting, A. Amini, S. Karaman, and D. Rus, "Deep orientation uncertainty learning based on a bingham loss," in *International Conference on Learning Representations*, 2020.

[25] V. Peretroukhin, M. Giamou, W. N. Greene, D. Rosen, J. Kelly, and N. Roy, "A Smooth Representation of Belief over SO(3) for Deep Rotation Learning with Uncertainty," in *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020.

[26] D. Choukroun, I. Y. Bar-Itzhack, and Y. Oshman, "Novel quaternion kalman filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 174–190, 2006.

[27] T. Yang, P. Mehta, and S. P. Meyn, "Feedback particle filter," *IEEE Transactions on Automatic Control*, vol. 58, pp. 2465–2480, 2013.

[28] S. Peng and H. Zhang, "Wong–zakai approximation for stochastic differential equations driven by g -brownian motion," *Journal of Theoretical Probability*, pp. 1–16, 2020.

[29] K. Berntorp, "Comparison of gain function approximation methods in the feedback particle filter," *2018 21st International Conference on Information Fusion*, pp. 123–130, 2018.

[30] A. W. Moore, "An introductory tutorial on kd-trees," 1991.