Quaternion EKF

Albert Author[1] and Bernard D. Researcher[2]

***Abstract*— Pose estimation has long been a subject of interest to researchers**

This paper presents an EKF based pose estimation and tracking for situations where direct sensing is not feasible. We expand on the work of Fathian and Dani, [1] and [2], by using their vision based pose and velocity estimation as remote sensing measurements to our EKF model.

Short sentences on what we will present

I. INTRODUCTION

Similar to Abstract but in past tense and in more detail

In this paper, we combine QUEST and VEST with Kalman Filter.

A. Background

The Kalman Filter (KF) was formally introduced in the summer of 1960 by Rudolf E. Kalman where, he formulated the state-space representation of dynamical systems [3].

B. Notation

Vectors are represented by normal font variables, i.e. a, x, y, z. Matrices are represented by UPPERCASE letters, i.e. $\mathbf{A}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}$. *Approximation* vectors are denoted with a *tilde* i.e. $\widetilde{a}, \widetilde{x}, \widetilde{y}, \widetilde{z}$. We define approximate function representation in 3 and 4 as the original function with added bias and uniform distribution noise. This form of representation is used to express measurements that are used as approximate control input to the system. This is an essential part the EKF implementation, which we describe in detail in later sections. *Estimation* vectors are represented with a *hat*, i.e. $\widehat{a}, \widehat{x}, \widehat{Y}, \widehat{Z}$.

C. Paper Structure

II. OBSERVATION AND ESTIMATION MODELS

A. Inertial Model

We start with a basic linear continuous-time state-space representation of a dynamical system and derive

For this paper, we used linear and angular velocities obtained from as known input to the system, where we define $u \in \mathbb{R}^6$ as,

$$u^T := \left[v^T \omega^T\right], \tag{1}$$

$$u_B = \begin{bmatrix} C_B^I \\ C_B^I \end{bmatrix} u, \tag{2}$$

where matrix C_B^I represents the orthanormal rotation from *inertial frame*, I, to robots *body frame*, B.

$$\widetilde{v} = v + n_v + b_v, \tag{3}$$

$$\widetilde{\omega} = \omega + n_\omega + b_\omega. \tag{4}$$

where n and b represent a uniform distribution or Brownian noise and bias from the measurement. Both n and b belong to $\mathbb{R}^3$

B. Linearization

C. Discretization

D. QUEST

E. VEST

F. Observation State Definition

The measurement state definition is defined by linear position, r, linear velocity, v, angular velocity, ω, and angular orientation in the quaternion space, q. The observation state vector, z, is defined as the following:

$$z^T := \left[r^T \; v^T \; \omega^T \; q_{xyzw}^T\right], \tag{5}$$

Where the observed the quaternion state is used to compute the corresponding state rotation matrix, C_B^I. It is important to note that the observation data for position and orientation are treated as *estimation groundtruth* (Keep?????) and velocity values are treated as *control inputs* to the system. Both estimation groundtruth and control inputs carry *process and observation noise*, respectively.

G. EKF Model

We deployed a modified EKF filter that uses *Quaternion* for representing rotations. We explain these modifications in detail in the *Quaternion Algebra*. We start with a nonlinear continuous-time system model described by,

$$\dot{x} = f(x, u), \tag{6}$$

$$y = h(x, u). \tag{7}$$

Where $f()$ represents the *process* model and $h()$ represent the *observation* model. Variables ω_f and ω_h represent the *process* and *observation* noise, respectively. Vector u represents input to the system and $u \in \mathbb{R}^6$. Vector y represents the system output and $y \sim\in \mathbb{R}^6$.

*This work was not supported by any organization

[1]Albert Author is with Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, 7500 AE Enschede, The Netherlands `albert.author@papercept.net`

[2]Bernard D. Researcheris with the Department of Electrical Engineering, Wright State University, Dayton, OH 45435, USA `b.d.researcher@ieee.org`

III. QUATERNION ALGEBRA

Quaternion space is a non-minimal representation belonging to *SO(3)* Lie group.

A. Unit Quaternion

Moreover, the quaternion term from the dataset has *four terms* with *xyzw* format. Hamilton's quaternion defined by 3 perpendicular imaginary axes i, j, k with real scalars x, y, z and a real term w, which constraints other 3 dimension to a *unit magnitude*. Thus, the fourth term normalizes the vector's magnitude conveniently and preserves the 3D rotation (3 DOF). We define **Unit Hamiltonian** or **Unit Quaternion** as,

$$\begin{aligned} \mathbb{H}^1 \quad := \quad & \{q_{wxyz} = w + xi + yj + zk \ \in \mathbb{H} \ | \\ & w^2 + x^2 + y^2 + z^2 = 1\} \end{aligned} \tag{8}$$

where superscript 1 in $\mathbb{H}^1$ denotes a unit quaternion space with 4 terms. There are two equal representations for $\mathbb{H}^1$ subgroup; thus, we provide a concise definition and notation for both to avoid confusion. The the first representation is shown in 5 where the four terms of the quaternion are arranged in *wxyz* order and it is represented by q_{wxyz}. The second quaternion is arranged in *xyzw* format and is represented by q_{xyzw}. It is important to note the difference as both are used in our derivation and implementation.

$$q_{wxyz} = q_{xyzw} \ ; \quad q_{wxyz},\ q_{xyzw} \in \mathbb{H}^1 \tag{9}$$

B. Pure Quaternion

As previously mentioned, the three imaginary terms of the quaternion represent the 3D angles of interest in radians and the fourth dimension constraints the vector magnitude. Thus to avoid computational errors, in the prediction step, we use the unit quaternion where it only has its three imaginary terms, *xyz*. This quaternion space representation is defined by $\mathbb{H}^0$ and denoted by $q_x yz$ variables.

$$\begin{aligned} \mathbb{H}^0 \quad := \quad & \{q_{xyz} = xi + yj + zk \ \in \mathbb{H} \ | \\ & x, y, z \in \mathbb{R}\} \simeq \mathbb{R}^3 \end{aligned} \tag{10}$$

C. Exponential Map

For calculating incremental rotation in

Incremental rotation estimation using the skew-symmetric matrix obtained form the rotational rate vector and matrix exponential mapping function, [QEKF01]. Gamma, Γ, represents incremental

$$\Gamma_0 := \sum_{i=0}^{\infty} \frac{(\Delta t^{i+n})}{(i+n)} \omega^{\times i}, \tag{11}$$

Where $(.)^{\times}$ represents skew-symmetry matrix of a vector

D. Updating Quaternion State

$$q_{i+1} = \delta q_i \otimes \widehat{q}_i \tag{12}$$

E. Capturing Quaternion Error

We use the mapping function $\zeta(.)$ to calculate the quaternion state error from the error rotation vector, [QEKF01].

$$\delta q = \zeta(\delta\phi), \tag{13}$$

$$\zeta : v \rightarrow \zeta(v) = \begin{bmatrix} \sin(\frac{1}{2}\|v\|)\frac{v}{\|v\|} \\ \cos(\frac{1}{2}\|v\|) \end{bmatrix} \tag{14}$$

IV. SYSTEM MODELING AND LINEARIZATION

A. Nonlinear Discrete Model

The QEKF2 paper (State Estimation for a Humanoid Robot) has the following discrete nonlinear model.

$$\hat{r}^-_{k+1} = \hat{r}^+_k + \Delta t \hat{v}^+_k + \frac{\Delta t^2}{2}(\hat{C}^{+T}_k \hat{f}_k + g) \tag{16}$$

$$\hat{v}^-_{k+1} = \hat{v}^+_k + \Delta t(\hat{C}^{+T}_k \hat{f}_k + g) \tag{17}$$

$$\hat{q}^-_{k+1} = \exp(\Delta t \hat{\omega}_k) \otimes \hat{q}^+_k \tag{18}$$

$$\hat{p}^-_{k+1} = \hat{p}^+_{i,k} \tag{19}$$

$$\hat{b}^-_{f,k+1} = \hat{b}^+_{f,k} \tag{20}$$

$$\hat{b}^-_{\omega,k+1} = \hat{b}^+_{\omega,k} \tag{21}$$

$$\hat{z}^-_{k+1} = \hat{z}^+_{i,k} \tag{22}$$

V. EXTENDED KALMAN FILTER

The Kalman Filter algorithm is capable of handling minor nonlinearities due to measurement noise by forming approximate Gaussian distributions about the state estimate. We deploy linearization techniques to develop a more percise model and use the *Extended Kalman Filter* (EKF) for state estimation.

A. Estimation Step

The following is the standard EKF

$$\begin{aligned} \hat{\mathbf{x}}_t &= \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_t) \\ \hat{\mathbf{P}}_t &= \mathbf{F}(\mathbf{x}_{t-1}, \mathbf{u}_t)\mathbf{P}_{t-1}\mathbf{F}^T(\mathbf{x}_{t-1}, \mathbf{u}_t) + \mathbf{Q}_t \end{aligned} \tag{15}$$

B. Variable State Definition

In the estimation state, first we estimate current state based on *prior* knowledge or observations. $\hat{x}$ represents the system variable state estimation vector with 10×1 dimensions and tracks linear position, linear velocity and rotation in unit quaternion. $\hat{r}$ and $\hat{v}$ vectors represent linear position and velocity estimation vectors, respectively, and each belong to the $\mathbb{R}^3$ space. $\hat{q}_{wxyz}$ represents the rigid object orientation in unit quaternion and belongs to $\mathbb{H}^1$ space.

$$\hat{x}^T := \left\langle \hat{r}^T \ \hat{v}^T \ \hat{q}^T_{wxyz} \right\rangle \tag{16}$$

It is important to note that $\hat{x}$ is in essence the same as *a posteriori* belief states at k and *a priori* belief states at $k+1$. In writing they are often presented as if they are different variables but in reality and in implementation, a posteriori and *a priori* belief states are the same the state variables at different discrete time instances. The $-$ and $+$ superscripts represent prior and posterior or before-update and after-update states, respectively.

$$\hat{x}_k^+ := \hat{x}_{k+1}^- \tag{17}$$

The state, x, is defined by rigid object's global linear position $r \in \mathbb{R}^3$, velocity $v \in \mathbb{R}^3$, and orientation $q_{xyz} \in \mathbb{H}^0$. We provide a detailed description for quaternion space representation in section 3.

$$x^T := \langle r^T \ v^T \ q_{xyz}^T \rangle \tag{18}$$

Estimation residual is denoted by $\delta x \in \mathbb{R} + \mathbb{H}$ (How can define the space here??)

$$\delta x^T = [\delta r^T \ \delta v^T \ \delta \phi^T] \tag{19}$$

TODO: How was $\delta\phi^T$ obtained?

This spatial transform is carried out using an *extension of Euler's formula*

$$R_k = Rot_{from_q}\left(q_{xyzw}^T \right) \tag{20}$$

C. Incremental Rotation from Unit Quaternion

The rotation matrix $C_k = k$ or for simplicity C is the *Euler angle* repsentation of the state rotation calculated from q_{xyzw} obtained from state observation vector z, 5. $C \in \mathbb{R}^3$ and has 3×3 dimensions.

$$\mathbf{C} = \begin{bmatrix} 1-2s(q_j^2+q_k^2) & 2s(q_iq_j - q_kq_c) & 2s(q_iq_k + q_jq_c) \\ 2s(q_iq_j + q_kq_c) & 1-2s(q_i^2+q_k^2) & 2s(q_jq_k - q_iq_c) \\ 2s(q_iq_k - q_jq_c) & 2s(q_jq_k + q_iq_c) & 1-2s(q_i^2+q_j^2) \end{bmatrix} \tag{21}$$

What H currently is. H represents *observation model* and it is a 12×9 matrix.

$$H = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{bmatrix} \tag{22}$$

And what it should be, H represents *observation model* with 9×9 dimensions, .

$$H = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \tag{23}$$

L is 9×9 matrix.

$$L = \begin{bmatrix} -I & 0 & 0 \\ 0 & -C^T & 0 \\ 0 & 0 & -I \end{bmatrix} \tag{24}$$

F is 9×9 matrix.

$$F = \begin{bmatrix} I & \Delta t I & 0 \\ 0 & I & 0 \\ 0 & 0 & I - \Delta t \omega^\times \end{bmatrix} \tag{25}$$

D. State Estimation Implementation

The following is our current implementation for estimation or prediction step. This dynamic model is based on discrete linear model in QEK2 paper [4]. Note: the F and H matrices are not properly implemented?

$$\hat{\mathbf{x}} = Fx + B[u^T, w^T]^T \tag{26}$$

$$\begin{aligned} \hat{r}_{k+1}^- &= \hat{r}_k^+ + \Delta t \hat{v}_k^+ \\ \hat{v}_{k+1}^- &= \hat{v}_k^+ \\ \hat{q}_{wxyz,\ k+1}^- &= \tilde{q}_{k+1} \otimes \hat{q}_k^+ \end{aligned} \tag{27}$$

where $\tilde{q}_{k+1}$ represent the incremental rotation change based on previous orientation $\hat{C}_k^T$ and the measured angular velocity $\tilde{\omega}_{k+1}$.

$$\tilde{q}_{k+1} = \exp\left(\Delta t \ \hat{C}_k^T \cdot \tilde{\omega}_{k+1}\right) \tag{28}$$

E. Discrete Covariance Estimation Implementation

The following equations correspond to our EKF implementation.

$$\begin{aligned} \mathbf{Q}_{k+1} &= \Delta t \ \mathbf{F}\mathbf{L}\mathbf{Q}_c\mathbf{L}^T\mathbf{F}^T \\ \hat{\mathbf{P}}_{k+1} &= \mathbf{F}\hat{\mathbf{P}}_k\mathbf{F}^T + \mathbf{Q}_{k+1} \end{aligned} \tag{29}$$

Diagonal matrix Q has 9×9 dimensions and represent the process noise tolerance or innovation.

$$Q_c = \begin{bmatrix} \mathbf{Q}_r & 0 & 0 \\ 0 & \mathbf{Q}_v & 0 \\ 0 & 0 & \mathbf{Q}_q \end{bmatrix} \tag{30}$$

F. Continuous Model

The following continuous model take noise into account to linearize the model by assuming non-zero noise.

$$\dot{\delta r} = \delta v \tag{25}$$

$$\dot{\delta v} = -C^T f^\times \delta\phi - C^T \delta b_f - C^T w_f \tag{26}$$

$$\dot{\delta \phi} = -\omega^\times \delta\phi - \delta b_\omega - w_\omega \tag{27}$$

$$\cancel{\dot{\delta p} = C^T w_p} \tag{28}$$

$$\dot{\delta b_f} = w_{bf} \tag{29}$$

$$\dot{\delta b_\omega} = w_{b\omega} \tag{30}$$

$$\cancel{\dot{\delta \theta} = w_z} \tag{31}$$

G. Discrete Linear Model

The following depicts what was presented in QEKF2 paper, [4].

H. EKF Update Step

The following is the standard EKF update step formulation.

$$\begin{aligned} v_t &= z_t - h(x_t) \\ S_t &= H(x_t)\hat{P}_t H^T(x_t) + R_t \\ K_t &= \hat{P}_t H^T(x_t) S_t^{-1} \\ x_t &= \hat{x}_t + K_t v_t \\ P_t &= \left(I_4 - K_t H(x_t)\right)\hat{P}_t \end{aligned} \tag{31}$$

nsated
corre-
noise
$, \delta\theta]^T$
. The
form

$$
\begin{array}{c|ccccccc}
 & r & v & q & p & b1 & b2 & z \\
\hline
r & & & & & & & \\
v & & & & & & & \\
q & & & & & & & \\
p & & & & & & & \\
b1 & & & & & & & \\
b2 & & & & & & & \\
z & & & & & & &
\end{array}
\quad
F_k = \begin{pmatrix}
I & I\Delta t & 0 & 0 & 0 & 0 & 0 \\
0 & I & -C_k^T f_k^{\times} \Delta t & 0 & -C_k^T \Delta t & 0 & 0 \\
0 & 0 & I - \omega_k^{\times} \Delta t & 0 & 0 & -I\Delta t & 0 \\
0 & 0 & 0 & I & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & I & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & I & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & I
\end{pmatrix},
$$

r_xyz
--> v_xyz
q_xyz

$$
\begin{array}{ccc}
r & v & q
\end{array}
\quad
H_k = \begin{pmatrix}
-C_k & 0 & (C_k(\cancel{p_k} - r_k))^{\times} & C_k & 0 & 0 & 0 \\
0 & 0 & I & 0 & 0 & 0 & -C[q_k \otimes z_k^{-1}]
\end{pmatrix}
$$

I. Update Step Implementation

The following equations represent our QEKF implementation. Note: the F and H matrices are not properly implemented?

$$
\begin{aligned}
H_x &= H_{12\times 9} x_{k,9\times 1} \\
S &= H_x \hat{P} H_x^T + R \\
K &= \hat{P} H_x^T S^{-1} \\
y_{r,v,w} &= z_{r,v,w} - H_{x;r,v,w}^T \\
y_q &= \log(z_q \bigotimes x_q) \\
y &= [y_{r,v,w}, y_q]^T \\
x &= \hat{x} + Ky \\
x_q &= \mathrm{map}_{exp}(\Delta t (K y_{\omega})) \\
P^+ &= (I - KH)P^-(I-KH)^T + KRK^T
\end{aligned}
\tag{32}
$$

where H is a 12×9 matrix. x is a 9×1 matrix.

J. Abbreviations and Acronyms

Defib

K. Units

L. Equations

The equations
Note

M. Some Common Mistakes

VI. USING THE TEMPLATE

Use this sample docu

A. Headings, etc

Text heads organiz

B. Figures and Tables

Positioning Figure

TABLE I
AN EXAMPLE OF A TABLE

One	Two
Three	Four

We suggest

Fig. 1. Inductance ofd

VII. CONCLUSIONS

A conclu

APPENDIX

Appendixes should appear before the acknowledgment.

ACKNOWLEDGMENT

The preferr

REFERENCES

[1] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans, "Quest: A quaternion-based approach for camera motion estimation from minimal feature points," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 857–864, 2018.

[2] A. P. Dani, N. Gans, and W. E. Dixon, "Position-based visual servo control of leader-follower formation using image-based relative pose and relative velocity estimation," in *2009 American Control Conference*, pp. 5271–5276, IEEE, 2009.

[3] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, pp. 35–45, 03 1960.

[4] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal, "State estimation for a humanoid robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 952–958, IEEE, 2014.