2007 IEEE International Conference on Control and Automation
Guangzhou, CHINA - May 30 to June 1, 2007

Two Efficient Implementation Forms of Unscented Kalman Filter

Shaolin Lü, Lin Cai, Lu Ding and Jiabin Chen
School of Information Science and Technology
Beijing Institute of Technology, Beijing, China 100081
Email: leo20980@sohu.com

***Abstract*—Two efficient implementation forms of Unscented Kalman Filter are described in this paper. They respectively utilize Cholesky factors and modified Cholesky factors to update state covariance, which make them have good numerical stability. Through computational complexity analysis, it is shown that the implementation using Cholesky factors is slower than the standard implementation. Also is shown that in some cases the form using modified Cholesky factors is faster than the standard Unscented Kalman Filter.**

I. INTRODUCTION

Since Kalman Filter[1] was discovered, it has achieved great success in many applications. Because Kalman Filter is only optimal for linear system under Gaussian assumptions, the Extended Kalman Filter (EKF) [2] was developed. Though EKF is widely used in nonlinear estimation, it has two problems. First, there is a matrix inversion algorithm in standard EKF, which makes it numerically unstable. To solve this problem, Cholesky factor updating and modified Cholesky factor updating were introduced into the implementation of EKF[3], [4], [5]. Second, nonlinear system must be linearized before EKF is used. Thereupon, EKF only has one order accuracy and it can easily lead to divergence. To circumvent this difficulty, Julier et al.[6], [7] proposed the Unscented Transformation(UT) to propagate mean and covariance, which has at least two order accuracy. It has been demonstrated in many applications[6], [7], [8], [9] that the UKF has superior performance over the EKF. To make UKF have better numerical properties, R. van der Merwe et al.[8] brought forward an implementation form using Cholesky factor updating.

In this paper, the SR-UKF is reformulated by utilizing other state covariance update methods. It is pointed that the SR-UKF definitely introduces more square roots. Furthermore, the implementation form of UKF using modified Cholesky factors, which is called UDU^T UKF in this paper, is introduced. Because the UDU^T UKF utilizes scalar measurement update, it computes more sigma-points than other implementation per update. Computation complexity analysis of different implementations is presented in detail. It is shown that the SR-UKF must do more operations than the standard UKF. Besides, only based on the UDU^T UKF's computational complexity, it is hard to conclude whether it is faster or slower than the standard UKF. Therefore, which implementation is the fastest can only be determined by the individual systems. However, the UDU^T UKF is faster than the standard UKF in many applications, especially in which there are many linear scalar measurements in measurement vector, since it does not need to calculate sigma-points for linear measurement. In addition, both the SR-UKF and the UDU^T UKF bring better numerical stability which will guarantee positive definiteness of the state covariance.

The discrete time nonlinear filtering problem may be formulated as follows:

$$x_{k+1} = F(x_k, u_k) + q_k \qquad (dynamic\ model) \tag{1}$$

$$y_k = H(x_k) + r_k \qquad (measurement\ model) \tag{2}$$

where $x_k \in \mathbb{R}^{\mathcal{N}}$ represent the system state, u_k is a known exogenous input, $y_k \in \mathbb{R}^{\mathcal{P}}$ is the measurement signal. q_k is the noise process,and r_k is the additive measurement noise. It is assumed that the noise vectors q_k and r_k are zero mean and

$$E[q(i)q^T(j)] = \delta_{ij}Q(i) \tag{3}$$

$$E[r(i)r^T(j)] = \delta_{ij}R(i) \tag{4}$$

$$E[q(i)r^T(j)] = 0, \quad \forall i, j \tag{5}$$

where it is also assumed that $Q \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$,$R \in \mathbb{R}^{\mathcal{P}\times\mathcal{P}}$ and R is a diagonal matrix. If R is not diagonal, it can be diagonalized by linear transformation, which is called whitening observation errors[2], [5]. The UKF utilizes sigma-points approach to recursively update the mean and covariance of the system state.

The organization of this paper proceeds as follows. In the next section, the unscented Kalman filter will be reviewed briefly and its computational complexity is analyzed. In section III, the SR-UKF will be reformulated. The UDU^T UKF will be introduced in section IV. The conclusion closes the paper.

II. THE UNSCENTED KALMAN FILTER

Julier et al[6], [7] introduced Unscented Transformation into nonlinear estimation. It approximates a probability by utilizing a deterministic sampling approach to calculate sigma-points. With these points and their respective weights, at least two order statistical information about the distribution can be captured. Then, the UKF propagates these points through the true nonlinear nonlinearity without approximation. Finally, a weighed mean and covariance can be got.

The points are chosen as follows

$$\mathcal{X}_{0,k} = \hat{x}_k \tag{6}$$

$$W_0 = \kappa/(\mathcal{N}+\kappa) \tag{7}$$

$$\mathcal{X}_{i,k} = \hat{x}_k - \sqrt{\mathcal{N}+\kappa}S_{i,k} \tag{8}$$

$$W_i = \frac{1}{2(\mathcal{N}+\kappa)} \tag{9}$$

$$\mathcal{X}_{i+n,k} = \hat{x}_k + \sqrt{\mathcal{N}+\kappa}S_{i,k} \tag{10}$$

$$W_{i+n} = \frac{1}{2(\mathcal{N}+\kappa)} \tag{11}$$

1-4244-0818-0/07/$20.00 © 2007 IEEE

where W_i is the weight of the ith sample point, $P_k = S_k S_k^T$ means estimate covariance of x_k, S_k is a triangular square matrix, and $S_{i,k}$ is the ith column of S_k. And then, $\lambda = \sqrt{\mathcal{N}+\kappa}$ is also defined. κ provides freedom to tune the high order moments of the approximation. When $\hat{x}_k$ is assumed Gaussian, κ can be selected to satisfy $\mathcal{N}+\kappa = 3$.

A. Algorithm Implementation

The standard implementation of UKF is given as followed Initialization:

$$\hat{x}_0 = E(x_0) \quad P_0 = E[(x_0 - \hat{x}_0)(x_0 - \hat{x_0})^T] \tag{12}$$

State Prediction:

$$S_{k-1} S_{k-1}^T = P_{k-1} \tag{13}$$

$$\mathcal{X}_{k-1} = [\hat{x}_{k-1} - \lambda S_{k-1}, \hat{x}_{k-1}, \hat{x}_{k-1} + \lambda S_{k-1}] \tag{14}$$

$$\mathcal{X}_{k|k-1} = F[x_{k-1}, u_{k-1}] \tag{15}$$

$$\hat{x}_k^- = \sum_{i=0}^{2\mathcal{N}} W_i \mathcal{X}_{i,k|k-1} \tag{16}$$

$$P_k^- = \sum_{i=0}^{2\mathcal{N}} W_i [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{X}_{i,k|k-1} - \hat{x}_k^-]^T + Q \tag{17}$$

Measurement Update:

$$\mathcal{Y}_{k|k-1} = H[\mathcal{X}_{k|k-1}] \tag{18}$$

$$\hat{y}_k^- = \sum_{i=0}^{2\mathcal{N}} W_i \mathcal{Y}_{i,k|k-1} \tag{19}$$

$$P_{y_k y_k} = \sum_{i=0}^{2\mathcal{N}} W_i [\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T + R \tag{20}$$

$$P_{x_k y_k} = \sum_{i=0}^{2\mathcal{N}} W_i [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T \tag{21}$$

$$K_k = P_{x_k y_k} P_{y_k y_k}^{-1} \tag{22}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (y_k - \hat{y}_k^-) \tag{23}$$

$$P_k = P_k^- - K_k P_{y_k y_k} K_k^T \tag{24}$$

B. Computational Complexity Analysis

The major disadvantage of the standard implementation is that the computation of P_k in Eqn. 24 can yield a computationally nonpositive result. The computational complexity of this implementation is analyzed below.

1) State Prediction: First, Cholesky decomposition of P_{k-1} requires about $\frac{1}{6}\mathcal{N}^3$ flops, with also $\mathcal{N}$ square roots. Second, calculating $\mathcal{X}_{k-1}$ approximately need $\mathcal{N}^2$ flops. Third, x_k^- calculation has to execute dynamic nonlinear mapping $2\mathcal{N}+1$ times. The last but not the least, the calculation of priori state and its covariance requires about $2\mathcal{N}^3 + 6\mathcal{N}^2$ flops. To summarize, state prediction of the standard UKF requires about $\frac{13}{6}\mathcal{N}^3 + 7\mathcal{N}^2$ flops and $2\mathcal{N}+1$ dynamic nonlinear mapping, with also $\mathcal{N}$ square roots.

2) Measurement Update: Calculating $\mathcal{Y}_{k|k-1}$ need perform measurement nonlinear mapping $2\mathcal{N}+1$ times. Measurement prediction requires $2\mathcal{N}\mathcal{P}$ flops. Calculating $P_{y_k y_k}$ will approximately cost $2(\mathcal{N}\mathcal{P}^2 + \mathcal{N}\mathcal{P} + \mathcal{P}^2)$ flops. If intermediate vectors are used to store $\mathcal{X}_{i,k|k-1} - \hat{x}_k^-$ and $\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-$, computation of $P_{x_k y_k}$ will only cost $2(\mathcal{N}+1)\mathcal{N}\mathcal{P}$ flops. For inverting P_{yy}^{-1}, operations count using matrix inverse algorithm given in [10] is about $\mathcal{P}^3$. On this account, operation count of calculating K_k is about $\mathcal{P}^3 + 2\mathcal{N}\mathcal{P}^2$. About $2\mathcal{N}\mathcal{P}$ flops are performed to compute $\hat{x}_k$. Finally, calculation of P_k will need $(2\mathcal{P}+1)\mathcal{N}^2 + 2\mathcal{N}\mathcal{P}^2$ operations. In a word, measurement update need cost about $\mathcal{P}^3 + 2\mathcal{P}\mathcal{N}^2 + 8\mathcal{N}\mathcal{P}^2 + 2\mathcal{P}^2 + \mathcal{N}^2$ flops and perform measurement nonlinear mapping $2\mathcal{N}+1$ times.

According to the analysis results above, it is clear that the standard UKF is a process of $\mathcal{O}(\frac{13}{6}\mathcal{N}^3 + 8\mathcal{N}\mathcal{P}^2 + 2\mathcal{P}\mathcal{N}^2 + 8\mathcal{N}^2 + \mathcal{P}^3)$ operations. It also can be concluded that the most computationally expensive function in the standard UKF implementation is updating covariances.

III. THE SQUARE-ROOT UNSCENTED KALMAN FILTER

There are many forms of square-root Kalman filter which use Cholesky factors of state covariance[2], [3], [4]. None of them is a process of $\mathcal{O}(\mathcal{N}^2)$ operations.

R. van der Merwe et al.[8] proposed that square-roots of state covariance could be updated in UKF. They utilized QR decomposition and cholesky factor updating

$$M = \underbrace{[\sqrt{W_i} \sum_{i=1}^{2\mathcal{N}} [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-] \ \sqrt{Q}]}_{2\mathcal{N}} \tag{25}$$

$$[Q, R] = qr(M^T) \tag{26}$$

$$S_k^- = \widetilde{R}^T \tag{27}$$

$$S_k^- = cholupdate\{S_k^-, \mathcal{X}_{0,k|k-1} - \hat{x}_k^-, W_0\} \tag{28}$$

where qr represents QR decomposition, $cholupdate$, which is a library function in Matlab, represents rank 1 Cholesky update and $\widetilde{R}$ is the upper triangular part of R. However, this algorithm is suspicious. It assumes that $\widetilde{R}^T \widetilde{R} = \sum_{i=1}^{2\mathcal{N}} W_i [\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{X}_{i,k|k-1} - \hat{x}_k^-]^T + Q$. The equation is correct only in some special cases.

In the stage of state prediction, there are at least four methods to update square-roots: Givens Rotations, Householder algorithm[1], rank 1 Cholesky update and direct method. Using Givens rotation is a process of $\mathcal{N}^3$ operations with $\frac{1}{2}\mathcal{N}(2\mathcal{N}+1)$ square roots. Householder algorithm' complexity is much more than $\mathcal{O}(\mathcal{N}^2)$. When rank 1 Cholesky update is used, it treats the square root of Q as original S and serially uses vectors $(\mathcal{X}_{i,k|k-1} - \hat{x}_k^-)$ to update S. Because each vector update is $\mathcal{O}(\mathcal{N}^2)$, using rank 1 Cholesky update needs at least $2\mathcal{N}^3$ operations. The most efficient method is direct method. In this stage, the Unscented Transformation can guarantee positive definiteness of state covariance. What is needed is that after been computed, the priori state covariance is decomposed it into Cholesky factor. This paper recommends direct method.

[1] See[2]for the two methods' theoretical and implementation details

In the stage of measurement update, it is better to compute $P_{y_k y_k}$ directly and decompose it. Finally, Givens rotations, which can guarantee positive definiteness of state covariance, can be utilized to update S_k.

A. Algorithm Implementation

The square-root UKF algorithm for nonlinear estimation is listed as follows
Initialization:

$$\hat{x}_0 = E(x_0) \tag{29}$$

$$S_0 S_0^T = E[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T] \tag{30}$$

State Prediction:

$$\mathcal{X}_{k-1} = [\hat{x}_{k-1} - \lambda S_{k-1}, \hat{x}_{k-1}, \hat{x}_{k-1} + \lambda S_{k-1}] \tag{31}$$

$$\mathcal{X}_{k|k-1} = F[\mathcal{X}_{k-1}, u_{k-1}] \tag{32}$$

$$\hat{x}_k^- = \sum_{i=0}^{2\mathcal{N}} W_i \mathcal{X}_{i,k|k-1} \tag{33}$$

$$S_k^- S_k^{-T} = P_k^- = \sum_{i=0}^{2\mathcal{N}} W_i[\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{X}_{i,k|k-1} - \hat{x}_k^-]^T + Q \tag{34}$$

Measurement update:

$$\mathcal{Y}_{k|k-1} = H[\mathcal{X}_{k|k-1}] \tag{35}$$

$$\hat{y}_k^- = \sum_{i=0}^{2\mathcal{N}} W_i \mathcal{Y}_{i,k|k-1} \tag{36}$$

$$S_y S_y^T = P_{y_k y_k} = \sum_{i=0}^{2\mathcal{N}} W_i[\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T + R \tag{37}$$

$$P_{x_k y_k} = \sum_{i=0}^{2\mathcal{N}} W_i[\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{Y}_{i,k|k-1} - \hat{y}_k^-]^T \tag{38}$$

$$K_k = (P_{x_k y_k}/S_y^T)/S_y \tag{39}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - \hat{y}_k^-) \tag{40}$$

$$S_k S_k^T = S_k^- S_k^{-T} - K_k S_y S_y^T K_k^T \tag{41}$$

B. Computational Complexity Analysis

Since square-root forms of the UKF recursively update Cholesky factors of state covariance, they have better numerical stability. At the same time, they need more operations, especially square roots. For this reason, their computation burden becomes heavier than that of stand UKF. Fortunately, the program of computing square root of a number runs quickly because it is a library function provided by today's compiler languages themselves.

1) State Prediction: In state prediction, the computation complexity is the same as that of the standard UKF. The difference is the sequence of operations. The standard UKF use Cholesky decomposition and square roots at the start of state prediction, while the square-root UKF uses them in the end.

2) Measurement Update: Using Givens rotations, the procedure of computing S_k needs about $\frac{4}{3}\mathcal{N}^3 + 2\mathcal{P}\mathcal{N}^2$ flops and $\frac{1}{2}\mathcal{N}(\mathcal{N} + 2\mathcal{P} + 1)$ square roots. The program of solving eqn. (39) $K_k = (P_{x_k y_k}/S_y^T)/S_y$ is easy to code. Since S_y is a triangular square matrix, only "back substitutions" whose whole count is $2\mathcal{N}\mathcal{P}^2$ are needed to solve K_k. Other procedures in this stage are the same as the standard UKF. Through comparison, it is not hard to find that in this stage the square-root UKF is more complex than the standard UKF.

IV. THE UDU^T UNSCENTED KALMAN FILTER

The UDU^T Kalman filter was proposed and well developed by Bierman and Thornton [5]. It has well-known numerical robustness and requires no square roots. In the UDU^T Kalman filter, state covariance is decomposed by using modified Cholesky decomposition $P = UDU^T$, in which U is unit upper triangular and D is diagonal. The UDU^T Kalman filter recursively updates U and D. In this section, UDU^T method will be introduced into the implementation of UKF.

In state prediction, the UDU^T UKF directly computes P_k^- and decomposes it into modified Cholesky factors.

In measurement update, the UDU^T square-root UKF makes use of two powerful techniques which will be briefly reviewed below. The first is scalar measurement update[2]. Using this technique, scalar components of measurement are processed serially as scalar observations. Thereupon, matrix inversion or solving linear equations is not needed. Therefore, the algorithm has better numerical stability. The second technique is rank 1 modification of modified Cholesky factors[3]. If U and D are the original modified Cholesky factors of P, then they are updated by solution $\hat{U}\hat{D}\hat{U}^T = UDU^T + aKK^T$, in which a is a scalar and K is an $\mathcal{N}$ vector. This algorithm is only $\mathcal{O}(\mathcal{N}^2)$ per update.

A. Algorithm Implementation

The UDU^T Unscented Kalman Filter for nonlinear estimation is presented as follows.
Initialization:

$$\hat{x}_0 = E(x_0) \tag{42}$$

$$U_0 D_0 U_0^T = E[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T] \tag{43}$$

State Prediction:

$$S_{k-1} = U_{k-1}\sqrt{D_{k-1}} \tag{44}$$

$$\mathcal{X}_{k-1} = [\hat{x}_{k-1} - \lambda S_{k-1}, \hat{x}_{k-1}, \hat{x}_{k-1} + \lambda S_{k-1}] \tag{45}$$

$$\mathcal{X}_{k|k-1} = F[\mathcal{X}_{k-1}, u_{k-1}] \tag{46}$$

$$\hat{x}_k^- = \sum_{i=0}^{2\mathcal{N}} W_i \mathcal{X}_{i,k|k-1} \tag{47}$$

$$U_k^- D_k^- U_k^{-T} = \sum_{i=0}^{2\mathcal{N}} W_i[\mathcal{X}_{i,k|k-1} - \hat{x}_k^-][\mathcal{X}_{i,k|k-1} - \hat{x}_k^-]^T + Q \tag{48}$$

[2] See [5] for details

[3] In [11], there are several implementations of this algorithm

Measurement Update:

$$U_- = U_k^-, D_- = D_k^-, \hat{x}_- = x_k^- \tag{49}$$

For $j = 1 \ldots \mathcal{P}$, $\mathcal{P}$ is the dimension of the measurement vector,

$$S_j^- = U_- \sqrt{D_-} \tag{50}$$

$$\mathcal{X}_j = [\hat{x}_- - \lambda S_j^-, \hat{x}_-, \hat{x}_- + \lambda S_j^-] \tag{51}$$

$$\hat{y}_j = H(\hat{x}_-) \tag{52}$$

$$\mathcal{Y}_j = H(\mathcal{X}_j) \tag{53}$$

$$P_{y_j y_j} = \sum_{i=0}^{2\mathcal{N}} W_i [\mathcal{Y}_{i,j} - \hat{y}_j]^2 + R_j \tag{54}$$

$$K_j = P_{y_j y_j}^{-1} \sum_{i=1}^{2\mathcal{N}} W_i [\mathcal{Y}_{i,j} - \hat{y}_j][\mathcal{X}_{i,j} - x^-] \tag{55}$$

$$\hat{x}^+ = \hat{x}_- + K_j(y_j - \hat{y}_j) \tag{56}$$

$$U_+ D_+ U_+^T = U_- D_- U_-^T - P_{y_j y_j} K_j K_j^T \tag{57}$$

$$U_- = U_+, D_- = D_+, \hat{x}_- = \hat{x}^+ \tag{58}$$

After the recurrence is ended, let

$$U_k = U_-, D_k = D_-, \hat{x}_k = \hat{x}_- \tag{59}$$

It should be noted that no operations are required in the procedure of solving Eqn. 49 and Eqn. 58-59. They are listed here to make the recurrence clearer.

B. *Computational Complexity Analysis*

This algorithm has good numerical stability since it does not need to solve linear equations or inverse a state covariance matrix. Furthermore, because except modified Cholesky factors every variable in measurement cycle is a vector or scalar, it can be easily implemented.

The only shortcoming of this algorithm is that it must calculate sigma points for each scalar measurement update, which brings $\mathcal{PN}$ square roots. Fortunatley , for modern compiler languages performing $\mathcal{PN}$ square roots is not a heavy burden compared with other operations per update. Furthermore, on the circumstances that there are many linear scalar measurements which exist in measurement model, the UDU^T UKF will be faster and more efficient than other forms of UKF because it is unnecessary to calculate sigma-points for linear measurement.

1) State Prediction: In this stage, the UDU^T UKF is the same as the standard UKF in the computational complexity. Both of them require N square roots and an $\mathcal{O}(\frac{1}{6}\mathcal{N}^3)$ matrix factorization.

2) Measurement Update: Each scalar measurement update costs about $5\mathcal{N}^2$ flops and $\mathcal{N}$ square roots. In measurement update, the whole amount of operations is approximately $5\mathcal{PN}^2$ plus $\mathcal{PN}$ square roots. Only according to the mount expressed with alphabets, it is difficult to judge whether the UDU^T UKF is faster or slower than the standard UKF. Thus, practical computation speed must be tested according to each individual system.

V. Conclusion

In this paper, square-root forms of UKF are reformulated and the UDU^T form is introduced. Each computational complexity is analyzed and compared with that of the standard UKF. And then, it is shown that the SR-UKF is more complex than the standard UKF. In many applications especially there are many linear scalar measurements in measurement vector, the UDU^T UKF is faster and more stable than the standard UKF. Sometimes, slow speed comes with good numerical stability. Therefore, engineers must seek for a good balance between computation complexity and numerical stability.

Acknowledgment

This work was sponsored by Weapon Equipment Pre-studying Fund No.9140A09030606BQ01.

References

[1] R. E. Kalman, A New Approach to Linear Filtering and Prediction Probelms, ASME Journal of Basic Engineering, Vol.82, 1960, pp. 34-45

[2] M. S. Grewal, A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB*, 2nd Edition, John Wiley & Sons, Inc., 2001

[3] P. G. Kaminski, A. E. Bryson, JR., and S. F. Schimidt, Discrete Square Root Filtering: A Survey of Current Techniques, *IEEE Trans. on AC*, Vol. AC-16, No.6, December 1971, pp. 727-736

[4] N. A. Carlson, Fast Triangular Formulation of the Square Root Filter, *AIAA Journal*, Vol.11, No.9, 1973, pp. 1259-1265

[5] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, Academic Press, New York: 1977

[6] S.J. Julier, J. K. Uhlmann and H. F. Durrant-Whyte , "A New Approach for Filtering Nonlinear Systems", in *Proceedings of the American Control Conference*, June 1995, pp. 1628-1632

[7] S.J. Julier and J. K. Uhlmann,Unscented Filtering and Nonlinear Estimation, in *Proceedings of the IEEE*, vol. 92, No.3, March 2004, pp. 401-422.

[8] R. van der Merwe and E. A. Wan, "The Square-root Unscented Kalman Filter for State and Parameter-Estimation", in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing* , Vol.6, May 2001, pp. 3461-3464.

[9] J. L. Crassidis, F. L. Markley, Unscented Filtering for Spacecraft Attitude Estimation, *Journal of Guidance, Control and Dynamics*,Vol.26, No.4, July-August 2003, pp.536-542

[10] W. Press, S. Teukolsky, W. Vetterling , B. Flannery, *Numerical Recipes in C*, Cambridge University Press, 1992

[11] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, Methods for Modifying Matrix Factorizations, *Math. Comp.*, Vol.28, Number 126, 1974, pp. 505-535