원격 프로시저 호출 웹서비스로 칼만필터 구현

임재걸*, 두쾅부*
*동국대학교 컴퓨터공학과
e-mail : yim@dongguk.ac.kr, vudq@aiti.com.vn

Implementation of Kalman Filter as a Remote Procedure Call Webservice

Jaegeol Yim*, Vu Quang Du*
*Dept. of Computer Engineering, Dongguk University

요 약

The Kalman Filter is one of the most common techniques of tracking mobile user. Web service is a promising method of reusing programs. Therefore, this paper presents our implementation of Kalman Filter web service. There are three ways of implementing a web service system. Ours is Kalman filter as RPC web service.

Key Words: Kalman Filter; Web Service; Remote Procedure Call, Mobile

1. Introduction

Web service [1-3] is one of the techniques of reusing programs developed by other organizations on their proprietary platforms. In web service, a published program is described in WSDL (Web Services Description Language) [4] and registered on UDDI (Universal Description, Discovery and Integration)[5]. Web application developers access a UDDI and search for a program that provides the function they want. They can use a published web service by inserting a few instructions of invoking it in their applications. The application communicates with the service provider in SOAP (Simple Object Access Protocol)[6]. There are three ways to implement a web service system: Remote procedure calls (RPC), Service oriented architecture (SOA) and Representational state transfer (REST).

Tracking of a mobile user has many practical applications. Making use of tracking techniques, many practical systems have been built. One of the most common tracking techniques is the Kalman Filter. In this paper, we will implement Kalman filter as RPC web service.

2. Kalman filter review

The basic equation of Kalman Filter is:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

with a measurement $z \in R^m$ that is

$$z_k = Hx_k + v_k$$

Where

- x_k is the mean state estimate of the previous step $k-1$
- A is the $n \times n$ transition matrix relates to the state at the previous step $k-1$
- B is the $n \times l$ effective matrix, relates to the optional control input $u \in R^l$
- H is the predictive probability of measurement
- w_k and v_k represent the process and measurement noises. They are assumed to be indepent (with each other), white, and with normal probability distributions:

$$p(w) \sim N(0, Q)$$
$$p(v) \sim N(0, R)$$

In practice, the process noise covariance and measurement noise covariance matrices, transition matrix, and predictive matrix H might change with each time step or measurement, however here we assume they are constant. Refer to [7, 8] for more information about the Kalman Filter.

The Kalman Filter process can be divided into two phases: The first is the prediction phase, where the next state of the system is predicted, and the next is updating or correction phase, where the current state of the system is estimated. See below:

- Prediction phase

$$X_k^- = A_{k-1}X_{k-1} + B_kU_k$$
$$P_k^- = A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1}$$

- Correction phase

$$V_k = Y_k - H_kX_k^-$$
$$S_k = H_kP_k^-H_k^T + R_k$$

"This research was supported by 'Industry, Academy and Research Institute Co-development of Techniques' funded by the Small and Medium Business Administration (R00046281)

$$K_k = P_k^- H_k^T S_k^-$$
$$X_k = X_k^- + K_k V_k$$
$$P_k = P_k^- - K_k S_k K_k^T$$

Where

- X_k^- and P_k^- are the predicted mean and covariance of the state, respectively, on the time step k before seeing the measurement
- X_k and P_k are the estimated mean and covariance of the state, respectively, on time step k after seeing the measurement.
- Y_k is mean of the measurement residual on time step k
- V_k is the innovation or the measurement residual on time step k
- S_k is the measurement prediction covariance on the time step k
- K_k is the filter gain, which tells how much the predictions should be corrected on time step k

3. Kalman filter service

Fig. 2 illustrates the way Kalman Service works. There are two parts: Kalman Service on a network server, and Kalman Filter (KF) Client on a machine that is able to reach the server. The client reads data from an input file, parses it and sends to the server (yellow path below), then the server applies the KF process on the data and transfers the result to the output file.

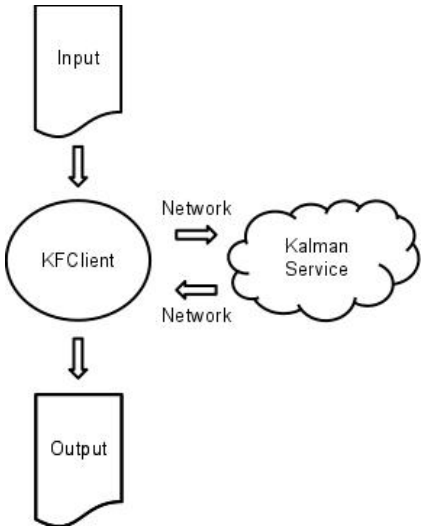

Fig. 1. Overview of Kalman Webservice

4. Detail design of system

Our Kalman Web service consists of three classes as shown in Fig. 2.

- Class Matrix defines matrix operators such as add (+) sub (-), multiply(*) and some other matrix operations (Inverse, Transpose)
- Class Kalman3D (three dimensions) is our implementation of the Kalman algorithm
- Class KFService implements RPC functions, that client can call to get service

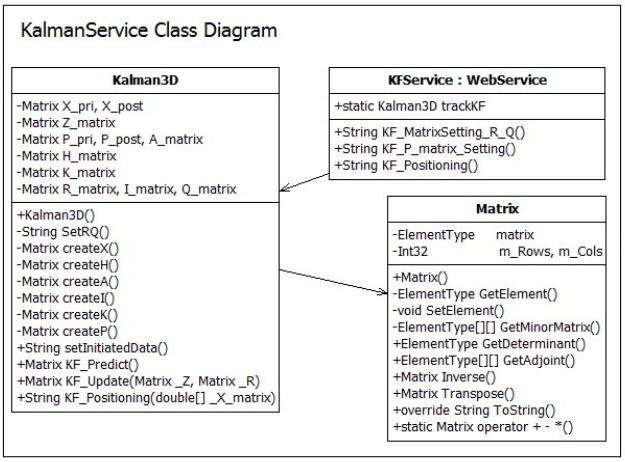

Fig. 2. Classes design of Kalman Webservice

The user interface of our client is shown in Fig. 3. We can define the error covariance matrix P_k, process noise covariance R, and measurement noise covariance Q. We also have to initialize the state vector X. We have reversed an area for the effective matrix B.

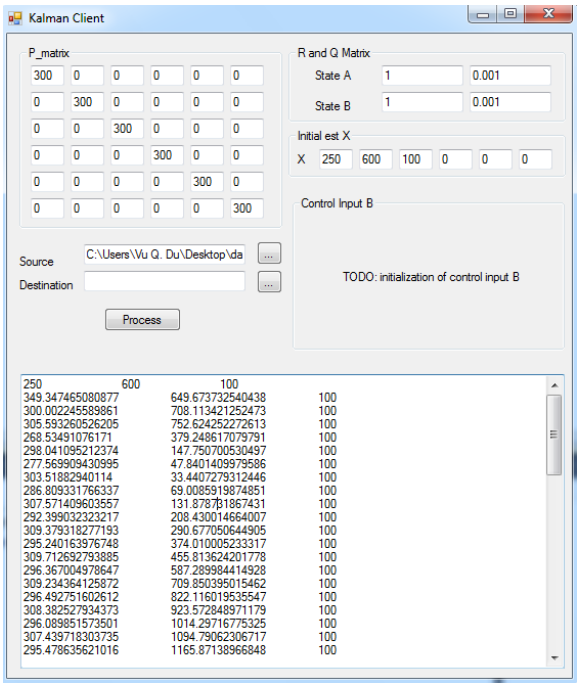

Fig. 3. The interface of KFClient

To use this service, a user must provide an input file. Each row in this file contains three numbers representing a measurement, x, y, z coordinates. These numbers are separated by a space-key. The format of output data file is similar to the format of the input file.

128.496596055288 142.769501543145 51.798710076950
169.715108556153 173.343747654352 49.246217559339
157.215003326886 233.844949697223 59.452205326719
…

5. Implementation

In this section, we describe our implementation of the Kalman Filter functions in C# language. The initialization procedure assigns initial values to all the global variables such as matrices X, P, K, H, I and A, as shown in Table 1.

<Table 1> The initialization process

```
Preparation:Initialize all preset data

public String setInitiatedData(double[][] _P_matrix, double[] _XInit)
{
    // Init X_matrix
    X_pri = createX(_XInit);
…

    // Init P_k
…
}
```

The prediction and the correction phases of the Kalman filter process is implemented as shown in Table 2 and 3, respectively.

<Table 2> Our implementation of the prediction phase

```
public Matrix KF_Predict() {
        // step (5)
        X_pri = A_matrix * X_pri;
        // step (6)
        P_pri = (A_matrix *P_post) * A_matrix.Transpose() + Q_matrix;
        return X_pri;
}
```

<Table 3> Our implementation of the correction phase

```
public Matrix KF_Update(Matrix _Z, Matrix _R){
    …
    //step (7)
    YM = _Z - H_matrix * X_pri;
    //step (8)
    S = H_matrix * P_pri * (H_matrix.Transpose()) + _R;
    //step (9)
    K_matrix = P_pri * (H_matrix.Transpose()) * (S.Inverse());
    //step (10)
    X_post = X_pri + K_matrix * YM;
    //step (11)
    P_post = P_pri - K_matrix*(H_matrix*P_pri);
    return X_post;
}
```

We have implemented *KF_MatrixSetting_R_Q(), KF_P_matrix_Setting()*, and *KF_Positioning()* web methods that can be used by application developers. *KF_Positioning()* is the main function of our implementation. It invokes KF_Predict, assigns the measurement to _Z, and invokes KF_Update.

Table 4: The web functions

```
public class KFService : System.Web.Services.WebService
{
    public static Kalman3D trackKF = new Kalman3D();

    //----------KF----------
    [WebMethod]
    public String KF_MatrixSetting_R_Q(ElementX _state)
    {
        return trackKF.SetRQ(_state);
    }
    [WebMethod]
    public String KF_P_matrix_Setting(double[][] _P_matrix, double[]
  _X_matrix)
    {
        return trackKF.setInitiatedData(_P_matrix, _X_matrix);
    }
    [WebMethod]
    public String KF_Positioning(double[] _X_matrix)
    {
        return trackKF.KF_Positioning(_X_matrix);
    }
}
```

5. Experiments

For the experiments, we have implemented a simple application program whose user interface is shown in Fig. 3. If a user types in values for P, Q and R, specifies the full path of the data file containing measurements, and click process button then the application invokes KF_Positioning web method and prints out the results on the textbox. A row of a data file represents a measurement consisting of x, y and z coordinates and a part of a sample data file is shown in Table 5.

Table 5. A sample input data file

250	1500	100
350	1550	100
…		
250	5000	100
350	5050	100

We have performed experiments of running the application with various datasets and parameter values. The first dataset we have used is shown in Table 5 and the widest zigzag line in Fig. 4 represents the dataset. We have run the program on the same dataset with various values of Q while R was fixed to 1. The results are plotted in Fig. 4 as narrow zigzag lines.

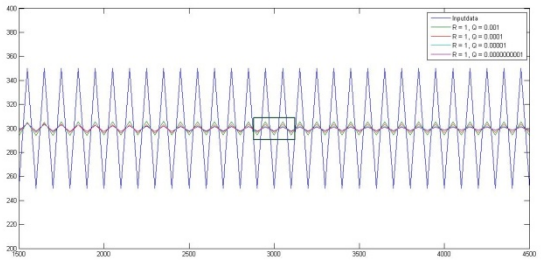

Fig. 4. Input and output with different values of R and Q

From the results shown in Fig. 4, we can notice that the

"This research was supported by 'Industry, Academy and Research Institute Co-development of Techniques' funded by the Small and Medium Business Administration (R00046281)."

width of the zigzag line gets narrower as the ratio of R/Q is bigger. However, recognizing the exact width of a zigzag line in Fig. 4 is hard, so we have magnified the small part represented by the rectangle in Fig. 4 as shown in Fig. 5.

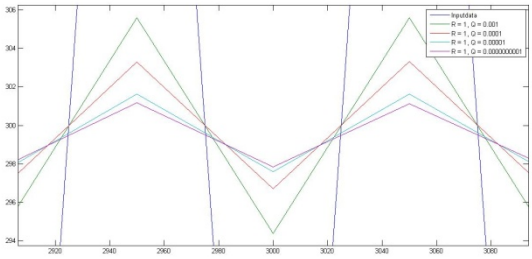

Fig. 5. A portion of Fig. 4 is magnified

We have tried a dataset obtained by the sinusoidal trigonometric function with random noise. Our test result is shown in Fig. 6.

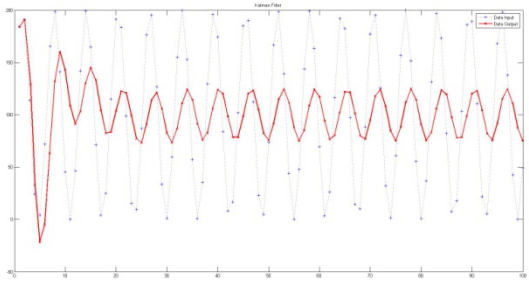

Fig. 6. Sinusoidal form Input and our output

Finally, we have tried datasets obtained by $y = ax + rand()$ and $y = rand() \times x + b$. Our test result for the former is shown in Fig. 7 and for the latter is shown in Fig. 8.

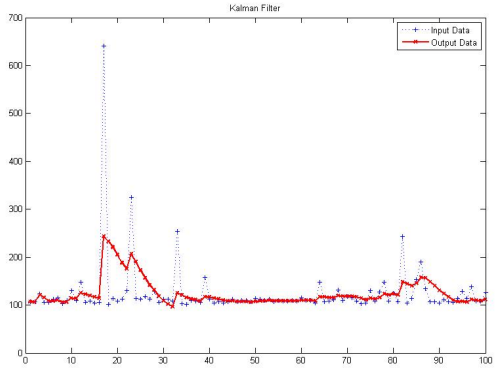

Fig. 7. Linear form Input $y = ax + rand()$ and our output

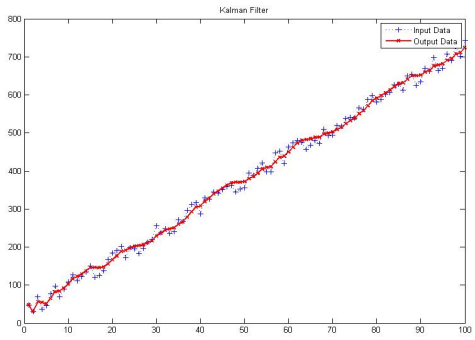

Fig. 8. Another Linear form Input $y = rand() \times x + b$ and our output

5. Conclusions

The Kalman filter process is widely used in various practical fields. Meanwhile, web service is believed to be one of the most efficient software development paradigms. Therefore, this paper introduced Kalman filter web service. We have also implemented a simple application making use of the web service and performed many experiments to show the web service is valid.

References

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, *Web Services: Concepts, Architectures and Applications* Springer, 2004.

[2] M. Stal, "Web Services: Beyond Component-Based Computing," Comm. ACM, vol. 55, no. 10, 2002.

[3] W. Vogels, "Web Services Are Not Distributed Objects," IEEE Internet Computing, vol. 7, no. 6, Nov./Dec. 2003.

[4] D. Booth et al., "Web Service Description Language (WSDL) Version 2.0," technical report, 2006.

[5] "UDDI technical white paper," W3C, 2000.

[6] D. Box et al., Simple Object Access Protocol (SOAP) 1.1, W3C note, 2000

[7] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Updated: Monday, July 24, 2006

[8] Improvement of Kalman Filters for WLAN based indoor tracking," Expert Systems With Applications, Vol. 37, No.1, Jan. 2010, pp.426-433