Technische Universität München
Institute of Communications and Navigation
Prof. Dr. sc. nat. Christoph Günther

Diplomarbeit

Clock Steering System Making Use of GPS Time Transfer

Yuan Lu

Munich, September 2006

Supervisor :
Dipl.-Ing. Alexandre Moudrak
Dr. Johann Furthner
Dipl.-Ing. Patrick Henkel

Diplomarbeit in
Institute of Communications and Navigation
of Technischen Universität München (TUM)
Titel : Clock Steering System Making Use of GPS Time Transfer
Autor : Yuan Lu

Yuan Lu
Karwendel Str. 27
81369 München
E-mail: luyuan@munihei.de

Acknowledgement

I would like to express my sincere gratitude to my supervisors Alexandre Moudrak, Dr. Johann Furthner and Patrick Henkel for their time, efforts and great patience for me during my diploma work. I would also like to thank Ulrich Grunert and Steffen Tölert for their support and help.

Moreover, I want to thank Prof. Dr. Christoph Günther for offering me the opportunity to work at his institute.

Then I would like to thank all the colleagues at the DLR for having been always ready to help me, my friends and family for supporting me throughout the work.

Contents

List of Figures

1 Introduction

1.1 Problem Statement

Accuracy and stability are the most important properties to judge the quality of a time scale. The kernel of a clock system is its frequency generator.

Several things will affect the quality of the time scale: System noise coming from the frequency source. Measurement noise from the interval counter.

By national time scale, large Cesium maser which can provide very accurate and stable frequency are applied. But in the small time keeping lab, usually, the frequency are generated by single or several commercial Cs masers. So it is necessary to steer it(them) to the reference time.

Time transfer usually use GPS common view or Two-way satellite time and frequency transfer (TWSTFT). The TWSTFT is a very expensive method to get the reference time.

So in this case, the UTC(DLR) time is generated with Cs maser 5071A and use GPS time transfer to steer it to the UTC(PTB). Kalman filter is implemented to make the time difference estimation and prediction in order to get the more accurate and stable time.

1.2 Motivation

The time laboratory of the German Aerospace Center generates for research and operational purposes its own time scale: UTC(DLR). To improve the accuracy of this time scale, it is necessary to continuously monitor the difference between UTC(DLR) and UTC(PTB) and correct the frequency.

Satellite navigation system GPS is used for time transfer.The time difference is calculated

from the output of the common view GPS receiver and the feed back of the PTB via FTP transfer.

$$\Delta T_{CV} = [UTC(PTB) - GPS\_Time] - [UTC(DLR) - GPS\_Time] \tag{1.1}$$

ΔT_{CV} Time difference between UTC(PTB) and UTC(DLR).

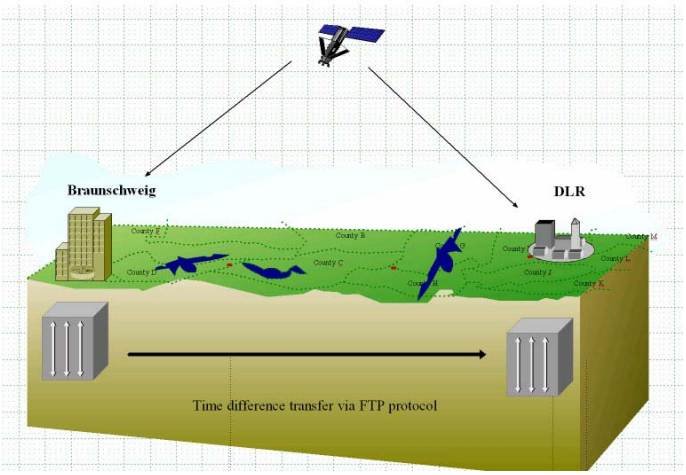

Figure 1.1: Making use of GPS time Transfer

After chosen the proper algorithm, a clock steering system is to be implemented in order to keep the UTC(DLR) steered to UTC(PTB).One data file contains these time differences will be used as the input of the steering system.

1.3 Out line of thesis

The outline of my paper is described as follows: In chapter 2 the related theoretical fundamentals are presented. In chapter 3 serval cases are studied to present some features of Kalman filter. Comparison of Kalman filter steering and INPL steering followed in chapter 4 to show the advantages of the Kalman-LQG method. In chapter 5 the implementation is described. Finally, my papers summarize in chapter 6.

2 Theoretical Fundamentals

In this chapter, we describe the theory of two methods used for clock steering - the Kalman filter and a method from the INPL, the National Physical Laboratory of Israel.

After introducing the system model of the Kalman filter, we deduce the fundamental properties of the Kalman filter including the Kalman gain and an estimate of the predicted covariance matrix. The state of our Kalman filter contains the time difference measurements. The fundamental trade-off between a small steering effort and minimum time offset is described.

The approach from the INPL is also a recursive method based on a simple measurement equation without any matrix inversion.

2.1 Kalman Filter

Preliminary work has been done by Thorvald N. Thiele and P. Swerling and was extended to the well-known Kalman filter by Rudolf E. Kalman. This recursive filter estimates a state vector of a time-discrete linear dynamic system.

The underlying system model is described by an observation and a state space equation. The observation equation comprises the mapping of the state vector to the measurements. The state space equation characterizes the relationship between the state of the previous and current epoch. Consequently, linear dynamics are modelled by the state space which occurs in both equations. The traditional Kalman filter operates in time domain and, thus, differs from frequency domain filtering. The application of the Kalman filter requires precise knowledge of the underlying noise processes for both measurements and state space.

2.1.1 General Form of Kalman Filter

The general model of the Kalman filter is given by

$$X_k = \phi X_{k-1} + \Gamma W_{k-1} \tag{2.1}$$

where X_k: system state vector at time t_k
ϕ: state transition matrix
Γ: system noise drive matrix
W_k: process noise vector which is assumed to be drawn from a zero mean multi-dimensional normal distribution with variance Q:

$$W_k \sim N(0, Q) \text{ [I]} \tag{2.2}$$

$$E \text{ [II]}[W_k W_i^T] = \begin{cases} Q, & i = k \\ 0, & i \neq k \end{cases} \tag{2.3}$$

The observation(measurement) Z_k of the true state X_k is made according to:

$$Z_k = H \cdot X_k + V_k \tag{2.4}$$

where H: observation model which map the true state space into the observed space
V_k: observation noise with zero mean and variance R: $V_k \sim N(0, R)$.

$$E[V_k V_i^T] = \begin{cases} R, & i = k \\ 0, & i \neq k \end{cases} \tag{2.5}$$

The general form of Kalman Filter can be written as:
Predict:

$$\hat{X}_{k/k-1} = \phi \cdot \hat{X}_{k-1} \tag{2.6}$$

$$P_{k/k-1} = \phi P_{k-1} \phi^T + \Gamma Q \Gamma^T \tag{2.7}$$

Update:

$$K = P_{k/k-1} H^T (H P_{k/k-1} H^T + R)^{-1} \tag{2.8}$$

[I] $X \sim N(\mu, \sigma^2)$ indicates that the variable X has the mean of μ and variance of σ^2
[II] $E(X)$ here indicates the mean value of variable X

$$\hat{X}_k = \hat{X}_{k/k-1} + K(Z_k - H\hat{X}_{k/k-1}) \tag{2.9}$$

$$P_k = (1 - KH)P_{k/k-1}(1 - KH)^T + KRK^T \tag{2.10}$$

where K: Kalman gain

$\hat{X}_{k/k-1}$: one-step-ahead prediction

$P_{k/k-1}$: one-step-ahead prediction error covariance

P_k: estimation error covariance

A follow chart of the Kalman filter for the k_{th} step is shown in Figure 2.1

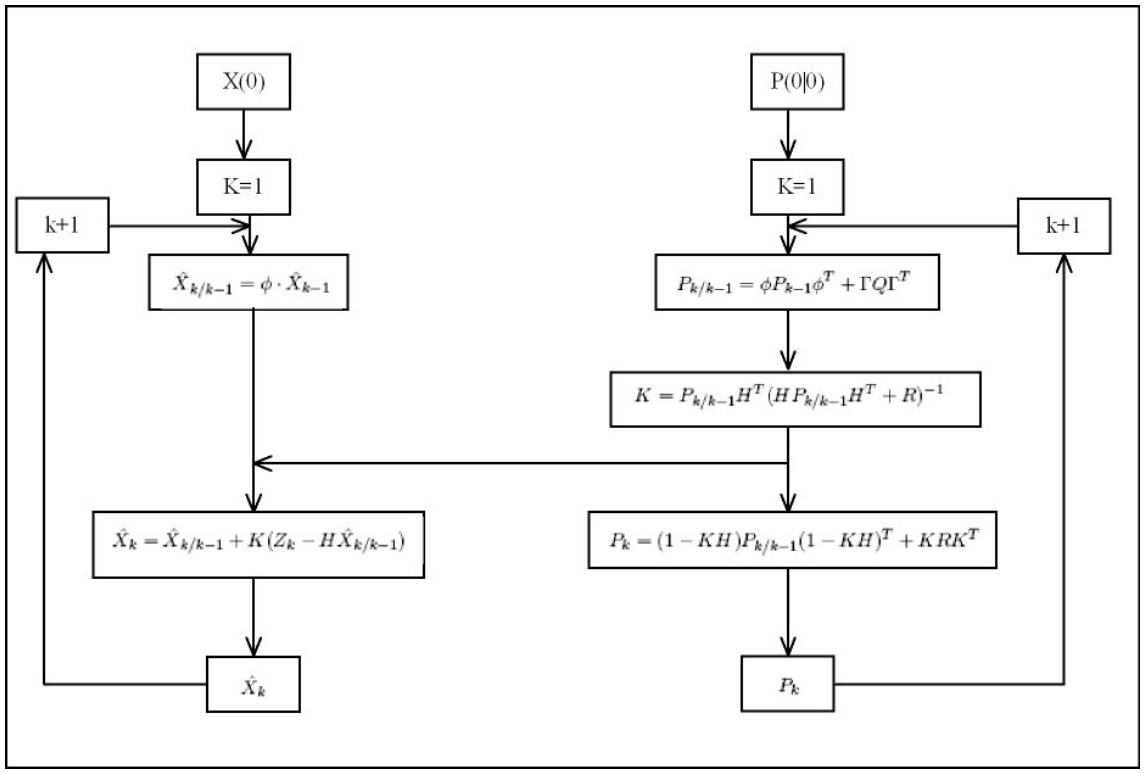

Figure 2.1: Kalman filter flow chart

2.1.2 Direct Proof of Kalman Filter [27]

The Kalman filter is a set of mathematical equations. In the following sections the predicted estimate covariance $P_{k/k-1}$, one step ahead prediction $\hat{X}_{k/k-1}$, status estimate $\hat{X}(k)$, updated estimate covariance P_k and the Kalman gain K (2.6 - 2.10) are deduced.

2.1.2.1 One Step Ahead Prediction $\hat{X}_{k/k-1}$

One Step Ahead Prediction is to estimate the status of kth step at the time of $(k-1)$th step. It also means to estimate the X_k for least variance according to the measurement $Z_1, Z_2, ..., Z_{k-1}$.

$$\hat{X}_{k/k-1} = E[X_k|Z_1 Z_2 \cdots Z_{k-1}] = E[(\phi X_{k-1} + \Gamma W_{k-1})|Z_1 Z_2 \cdots Z_{k-1}] \tag{2.11}$$

The symbol $E[X|Z_1 Z_2 \cdots Z_k]$ here means estimating X from historical measurements $Z_1, Z_2, \cdots, Z_k$.

Because of the linearity of the Least Variance Estimation,

$$\hat{X}_{k/k-1} = \phi E[X_{k-1}|Z_1 Z_2 \cdots Z_{k-1}] + \Gamma E[W_{k-1}|Z_1 Z_2 \cdots Z_{k-1}] \tag{2.12}$$

It's known from Equation (2.1) that W_{k-1} only affect X_k. So W_{k-1} is not correlated with $Z_1, Z_2, \cdots, Z_{k-1}$. And $E[W_{k-1}] = 0$.

$$E[W_{k-1}|Z_1 Z_2 \cdots Z_{k-1}] = 0 \tag{2.13}$$

And

$$E[X_{k-1}|Z_1 Z_2 \cdots Z_{k-1}] = \hat{X}_{k-1} \tag{2.14}$$

So equation 2.6 is obtained:

$$\hat{X}_{k/k-1} = \phi \hat{X}_{k-1} \tag{2.15}$$

2.1.2.2 Status Estimate

If use one-step-ahead prediction as the system state, it will cause an error:

$$\tilde{X}_{k/k-1} = X_k - \hat{X}_{k/k-1} \tag{2.16}$$

where $\hat{X}$ represents the estimate and $\tilde{X}$ describes the estimation error.

Then the estimated measurement would be:

$$\hat{Z}_{k/k-1} = H\hat{X}_{k/k-1} \tag{2.17}$$

The error between estimate and the real measurement is:

$$\tilde{Z}_{k/k-1} = Z_k - \hat{Z}_{k/k-1} = HX_k + V_k - H\hat{X}_{k/k-1} = H\tilde{X}_{k/k-1} + V_k \tag{2.18}$$

$\tilde{Z}_{k/k-1}$ contains the information of error $\tilde{X}_{k/k-1}$, if a proper weighting is applied to correct $\hat{X}_{k/k-1}$, the system status estimate $\hat{X}_k$ can be acquired.

Suppose K is the optimal weight:

$$\hat{X}_k = \hat{X}_{k/k-1} + K\tilde{Z}_{k/k-1} = \hat{X}_{k/k-1} + K(Z_k - \hat{Z}_{k/k-1}) = \hat{X}_{k/k-1} + K(Z_k - H\hat{X}_{k/k-1}) \tag{2.19}$$

2.1.2.3 Updated Estimate Covariance P_k

The definition of estimation error covariance is:

$$P_k = E[e_k e_k^T] = E[(X_k - \hat{X}_k)(X_k - \hat{X}_k)^T] \tag{2.20}$$

Substitute 2.9 into 2.20:

$$\begin{aligned}
P_k &= E\{[(X_k - \hat{X}_{k/k-1}) - K(Z_k - H\hat{X}_{k/k-1})][(X_k - \hat{X}_{k/k-1}) - K(Z_k - H\hat{X}_{k/k-1})]^T\} \\
&= E\{[(X_k - \hat{X}_{k/k-1}) - K(HX_k + V_k - H\hat{X}_{k/k-1})][(X_k - \hat{X}_{k/k-1}) - K(HX_k + V_k - H\hat{X}_{k/k-1})]^T\} \\
&= E\{[(X_k - \hat{X}_{k/k-1}) - KH(X_k - \hat{X}_{k/k-1}) - KV_k][(X_k - \hat{X}_{k/k-1}) - KH(X_k - \hat{X}_{k/k-1}) - KV_k]^T\} \\
&= E\{[(1 - KH)(X_k - \hat{X}_{k/k-1}) - KV_k][(1 - KH)(X_k - \hat{X}_{k/k-1}) - KV_k]^T\}
\end{aligned}$$

Because $X_k - \hat{X}_{k/k-1}$ is uncorrelated with V_k , $(AB)^T = B^T A^T$ and $E[V_k V_k^T] = R$:

$$\begin{aligned} P_k &= E\{[(1-KH)(X_k - \hat{X}_{k/k-1})(X_k - \hat{X}_{k/k-1})^T(1-KH)^T + KV_kV_k^TK^T]\} \\ &= (1-KH)P_{k/k-1}(1-KH)^T + KRK^T \end{aligned}$$

2.1.2.4 Kalman Gain K

The estimation of the Kalman filter comes both from the estimation which is based on the earlier steps and the measurement of the current step. Kalman gain K is the optimal weighting between these two parts to minimize the estimate error.

$$\begin{aligned} P_k &= (1-KH)P_{k/k-1}(1-KH)^T + KRK^T \\ &= (P_{k/k-1} - KHP_{k/k-1})(1-KH)^T + KRK^T \\ &= P_{k/k-1} - KHP_{k/k-1} - P_{k/k-1}H^TK^T + KHP_{k/k-1}H^TK^T + KRK^T \end{aligned}$$

According to the theorem of matrix calculus[10]:

$$\frac{\partial AB}{\partial A} = B^T \qquad \text{(A B must be square)} \tag{2.21}$$

$$\frac{\partial ACA^T}{\partial A} = 2AC \qquad \text{(C must be symmetric)} \tag{2.22}$$

and proceed to differentiate the P_k with respect to K.

$$\begin{aligned} \frac{\partial P_k}{\partial K} &= -(HP_{k/k-1})^T - P_{k/k-1}H^T + 2KHP_{k/k-1}H^T + 2KR \\ &= -P_{k/k-1}^TH^T - P_{k/k-1}H^T + 2KHP_{k/k-1}H^T + 2KR \end{aligned} \tag{2.23}$$

According to the following two properties of the symmetric matrix:

If X is a symmetric matrix, so is AXA^T for any matrix A. [APP. A]
If A and B are both symmetric, so is A+B. [24]

and because $P_{k/k-1} = \phi P_{k-1}\phi^T + \Gamma Q \Gamma^T$ (2.7), $P_{k/k-1}$ is a symmetric matrix.

Therefore the equation (2.23) can be rewritten as:

$$\frac{\partial P_k}{\partial K} = -2(P_{k/k-1}^T H^T) + 2K(HP_{k/k-1}H^T + R) \tag{2.24}$$

The optimal gain for Kalman filter is given from the following equation:

$$K = P_{k/k-1}^T H^T (HP_{k/k-1}H^T + R)^{-1} \tag{2.25}$$

2.1.2.5 Predicted Estimate Covariance $P_{k/k-1}$

The one-step-ahead prediction error covariance is defined as:

$$P_{k/k-1} = E[e_{k/k-1}e_{k/k-1}^T] = E[(X_k - \hat{X}_{k/k-1})(X_k - \hat{X}_{k/k-1})^T] \tag{2.26}$$

Substitute (2.6) into (2.26) and using equation (2.1):

$$\begin{aligned}
P_{k/k-1} &= E[(\phi X_{k-1} + \Gamma W_{k-1} - \phi \cdot \hat{X}_{k-1})(\phi X_{k-1} + \Gamma W_{k-1} - \phi \cdot \hat{X}_{k-1})^T] \\
&= E\{[\phi(X_{k-1} - \hat{X}_{k-1}) + \Gamma W_{k-1}][\phi(X_{k-1} - \hat{X}_{k-1}) + \Gamma W_{k-1}]^T\} \\
&= E[(\phi e_{k-1} + \Gamma W_{k-1})(\phi e_{k-1} + \Gamma W_{k-1})^T] \\
&= \phi \underbrace{E[e_{k-1}e_{k-1}^T]}_{P_{k-1}} \phi^T - \underbrace{\Gamma W_k e_{k-1}^T \phi^T}_{=0} - \underbrace{\phi e_{k-1} W_k^T \Gamma^T}_{=0} + \Gamma \underbrace{E[W_k W_k^T]}_{Q} \Gamma^T \\
&= \phi P_{k-1} \phi^T + \Gamma Q \Gamma^T
\end{aligned}$$

2.2 Linear Quadratic Gaussian Technique (LQG)

Steering policy affects the stability and accuracy of the time. Hard steering, which means to steer the time as soon as possible to the reference time, will reduce the stability. Soft steering will result a smoother time output. But at the same time it reduces the short term accuracy. So the target is to find an acceptable trade off between the short term accuracy and the stability.

2.2.1 LQG for smoothed output

LQG is a method of designing feed back control laws of linear system with additive Gaussian noise processes that minimize a given quadratic cost function. To get the better system stability, Linear Quadratic optimal control is chosen as the control strategy.

The discrete time system is described as:

$$x(t+1) = Ax(t) + Bu(t) \tag{2.27}$$

where u_t is the control element.

Because a large u can drive x fast to zero [5]. We want to choose $u(t)$ so that

x_n is 'small', this means we get the the good regulation or control
u_n is 'small', this means using small input effort

To make sure the steering and the offset are both minimized, the quadratic cost function J_u is defined as[12] [18]:

$$J(U) = \sum_{\tau=0}^{N-1} [x(\tau)^T Q x(\tau) + u(\tau)^T R u(\tau) + x(N)^T Q_f x(N)] \tag{2.28}$$

where state cost: $Q = Q^T \geq 0$, final state cost: $Q_f = Q_f^T \geq 0$, input matrices:$R = R^T \geq 0$.

Q, R set relative weights of state deviation and input usage and are assumed independent of τ.

LQG problem: In order to help give the linear approximation validity, the control vector $u_{opt}(n)$ is chosen such that the quadratic cost function J(U) is minimized.

2.2.2 Dynamic Programming Solution

To find the optimal u_k for the cost function J_U, one needs to calculate the J_U. But it's a recursive process. The solutions for this kind of problems are very natural but also very inefficient, because many identical recursive processes are repeat during the computation. [6] Dynamic programming solution gives an efficient, recursive method to solve LQR least-squares problem[5]. This method first breaks the problem into smaller subproblems, then solve the problems recursively. [21]

For t=0,1,...N define the value function $V_t : R^n$ by

$$V_t(z) = \min_{u(t),\ldots,u(N-1)} \sum_{\tau=t}^{N-1} [x(\tau)^T Q x(\tau) + u(\tau)^T R u(\tau) + x(N)^T Q_f x(N)] \tag{2.29}$$

where $V_t(z)$ is the minimum LQR cost starting from state $x(t)$ at time t

Let $x(t) = z$, $x(t+1) = Ax(t) + Bu(t)$

Because V_t is quadratic, $V_t(z) = z^T P_t z$, and $P_t = P_t^T \geq 0$

Dynamic programming usually takes one of two approaches: Top-down approach or Bottom-up approach. In this case Top-down approach is chosen. P_t can be found recursively, working backward from $t = N$

By final state cost: $V_N(z) = z^T P_N z$ where $P_N = Q_f$

$V_{t+1}(z)$ is the minimum cost starting from state $t+1$. So if starting from state t, the cost is:

$$V_t(z) = \min_w(z^T Q z + w^T R w + V_{t+1}(Az + Bw)) \tag{2.30}$$

Because $V_N(z) = z^T P_N z$ where $P_N = Q_f$, starting the Top-down approach:

$$V_{N-1}(z) = z^T Q z + \min_w(w^T R w + (Az + Bw)^T P_N (Az + Bw)) \tag{2.31}$$

The optimal control can be found by differentiating V_{N-1} with w:

$$2w^T R + 2(Az + Bw)^T P_N B = 0 \tag{2.32}$$

So optimal control w^* is:

$$w^* = -(R + B^T P_N B)^{-1} B^T P_N A z \tag{2.33}$$

and bring w^* back to equation 2.31:

$$\begin{aligned} V_{N-1}(z) &= z^T Q z + w^{*T} R w^* + (Az + Bw^*)^T P_N (Az + Bw^*) \\ &= z^T (Q + A^T P_N A - A^T P_N B (R + B^T P_N B)^{-1} B^T P_N A) z \end{aligned}$$

Because V_{N-1} is also quadratic: $V_{N-1}(z) = z^T P_{N-1} z$

$$P_{N-1} = Q + A^T P_N A - A^T P_N B (R + B^T P_N B)^{-1} B^T P_N A \tag{2.34}$$

This recursion works as well for all t:

$$P_{t-1} = Q + A^T P_t A - A^T P_t B (R + B^T P_t B)^{-1} B^T P_t A \tag{2.35}$$

With equation 2.35 $P_0, P_1, \cdots, P_N$ can all be found by recursion backward in time. This is so called Ricatti recursion[5].

The optimal control for the given cost function is give by the linear equation:

$$u(k) = -\hat{G}\hat{x}(k) \tag{2.36}$$

where the gain:

$$\hat{G} := -(R + B^T P_t B)^{-1} B^T P_t A \tag{2.37}$$

and P_t is the solution of the equation 2.35

2.3 INPL way of Steering

2.3.1 Description

The National Physical Laboratory of Israel (INPL) together with Time and Frequency Limited (TFL) and National Bureau of Standards of USA (NBS) has built a software clock to be used as the Israel national time base, UTC(INPL). The software clock is based on several commercial Cs clocks (Hewllet Packard-HP and TFL) whose outputs are routed sequentially through a TFL programmed switch into a time interval counter. The phase differences are processed in a personal computer using a procedure adopted from NBS to generate the software clock. The system also has an input from a common view GPS receiver for time comparisons. Steering of the software clock is possible from the PC keyboard. [3]

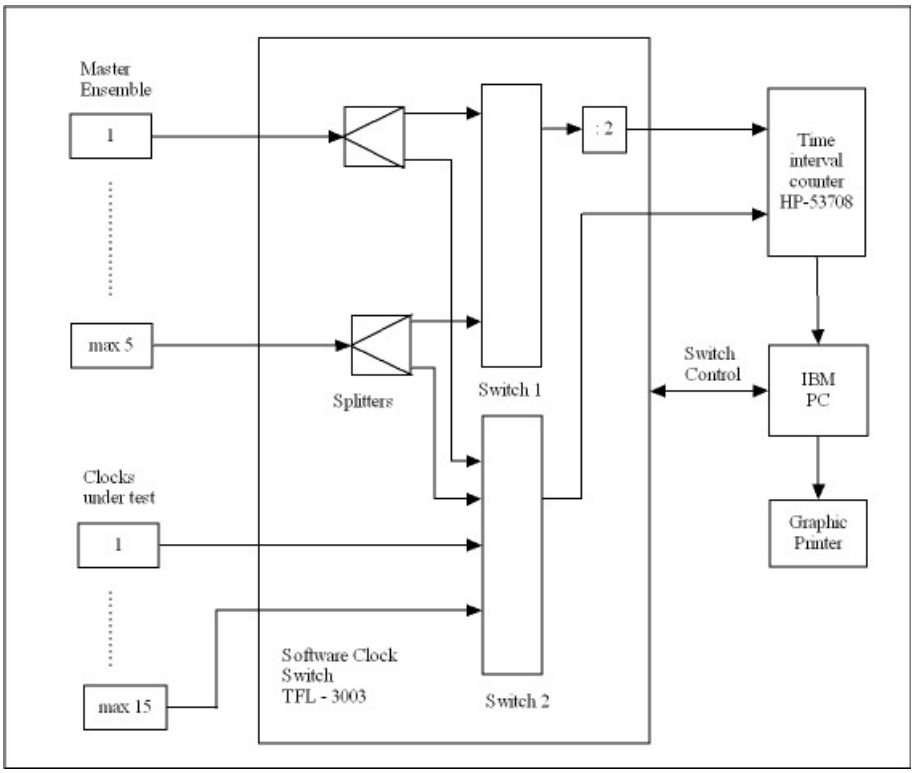

Figure 2.2: Block Diagram of the Software Clock System

2.3.2 Averaged Output

The software clock is generated from averaging all clocks connected to the system. The weights of the each clock is calculated dynamically.

Dynamic weighting is defined as:

$$E(t+T_1) = [SSD(t) + Nr * E(t)]/(N_r + 1) \tag{2.38}$$

$$W(T+T_1) = k/E(t+T_1) \tag{2.39}$$

where T_1: the time between iterations chosen in the present procedure to be 24 hours,
SSD: the squared second(the unit of time) difference of each clock from the software clock,
E: the filtered squared second(the unit of time) difference,
W: the weight to be used in the next iteration,
k: a dynamic normalization factor which keeps the sum of all W's equal to 1 and
N_r: the weight of exponential filter.

Expanding equation (2.38) :

$$E(t+T_1) = \frac{SSD(t) + Nr * E(t)}{1 + N_r} \tag{2.40}$$

$$\begin{aligned} E(t+2T_1) &= \frac{SSD(t+T_1) + Nr * E(t+T_1)}{1 + N_r} \\ &= \frac{SSD(t+T_1)}{1 + N_r} + \frac{Nr}{(1+N_r)^2}(SSD(t) + Nr * E(t)) \end{aligned} \tag{2.41}$$

$$
\begin{aligned}
E(t+3T_1) &= \frac{SSD(t+2T_1) + Nr * E(t+2T_1)}{1+N_r} \\
&= \frac{SSD(t+2T_1)}{1+N_r} + \frac{Nr}{1+N_r}\left(\frac{SSD(t)}{1+N_r} + \frac{Nr}{(1+N_r)^2}(SSD(t) + Nr * E(t))\right) \\
&= \frac{1}{(1+N_r)^1}SSD(t+2T_1) + \frac{Nr}{(1+N_r)^2}SSD(t+T_1) + \frac{Nr^2}{(1+N_r)^3}SSD(t) \\
&+ \frac{Nr^3}{(1+N_r)^3}E(t)
\end{aligned}
\tag{2.42}
$$

$$\vdots$$

$$
E(t+n\cdot T_1) = \left[\sum_{k=1}^{n} \frac{Nr^{k-1}}{(1+N_r)^k} SSD(t+(n-k)T_1)\right] + \left(\frac{Nr}{1+N_r}\right)^n \cdot E(t) \tag{2.43}
$$

Equation 2.43 shows the weights that each SSD measurement contributes to the clock weight for the next iteration.

Once the offsets Y_{io} from the software clock are computed[9], they are exponentially filtered with a dynamic time constant, Mi, which is related to the time interval for which Allan variance is minimum.[3]

where $\hat{Y}_{io}$ and $\hat{X}_{io}$ are the next predicted values of the frequency offset and phase respectively of the i^{th} clock.

$$\hat{Y}_{io}(t+T_1) = [Y_{io}(t+T_1) + Mi * \hat{Y}_{io}(t)]/(Mi+1) \tag{2.44}$$

$$\hat{X}_{io}(t+T_1) = X_{io}(t) + \hat{Y}_{io}(t) * T_1 \tag{2.45}$$

where X_{io} is the final phase difference from the software clock.

$$X_{io} = \sum_{j=1}^{n} W_j(t-T_1) * [\hat{X}_{jo}(t) - X_{ji}(t)] \tag{2.46}$$

2.3.3 Steering

The task of the steering control loop is to keep the system steered phase and fequency as close as possible to UTC. The steering form in the reference [3] is:

$$Y_s(t+T) = \{m * Y_s(t) - [X_{sutc}(t) - X_{sutc}(t-T)]/T\} * (m+1) - l * X_{sutc}/T \quad (2.47)$$

In the steered system, X_{iutc}, the phase of clock i with respect to UTC is:

$$X_{utc} = X_{iutc} - X_{io} - X_{os} \quad (2.48)$$

$$X_{is} = X_{io} + X_{os} \quad (2.49)$$

And it is mentioned that the equation corresponds to a exponential filter. A filter like that has the form of:

$$F(k+1) = \{m * F(k) + G(k)\}/(m+1) \quad (2.50)$$

where F_k is the k_{th} state of the exponential filter.
G_k is the k_{th} update to the system

It is supposed that the author wanted to write equation 2.47 slightly different and suggest that the following expression to be consistent to [9] [25]:

$$Y_s(t+T) = \{m * Y_s(t) + [X_{sutc}(t) - X_{sutc}(t-T)]/T\}/(m+1) + l * X_{sutc}/T \quad (2.51)$$

where X_{os} is the steering phase added to the software phase.
X_{sutc} is the steered output with respect to UTC.
Y_s is the changing rate of the X_{os}.

2.4 Moving average

Moving average is a common method to smooth out the short term fluctuations in stochastic processes. It's also applied to analyse the stock prices, returns or trading volumes. Moving average means one family of similar statistical techniques. Most commonly used moving average methods are: Single Moving Average (SMA), Weighted Moving Average (WMA) and Exponential Moving Average (EMA). [23]

Single Moving Average:

$$SMA = \frac{p_1 + p_2 + \cdots + p_n}{n} \tag{2.52}$$

Weighted Moving Average:

$$WMA = \frac{np_1 + (n-1)p_2 + \cdots + 2p_{n-1} + p_n}{n + (n-1) + \cdots + 2 + 1} \tag{2.53}$$

Exponential Moving Average:

$$EMA = \frac{p_1 + (1-a)p_2 + (1-a)^2 p_3 + (1-a)^3 p_4 \cdots}{1 + (1-a) + (1-a)^2 + (1-a)^3 + \cdots} \tag{2.54}$$

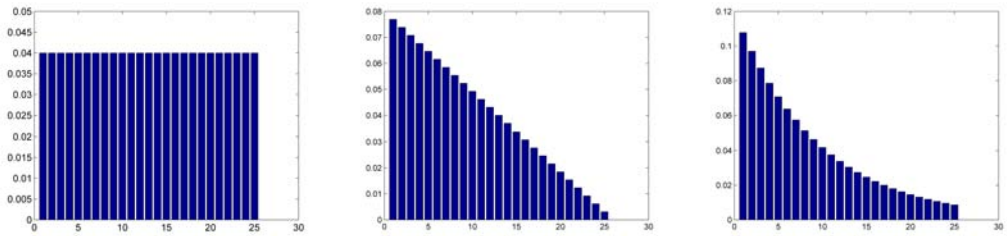

Figure 2.3: The weights of Moving average for 25 values SMA(left), WMA(middle) and EMA(right)

2.5 Allan Variance

Allan Variance is named after David W.Allan and describes the stability of a frequency source or oscillator. The main benefits of the Allan variance compared to the classical standard deviation is its convergence properties. An infinite standard deviation still corresponds to a finite Allan deviation. Allan variance are used to describe both short and long term properties of the clock.

If there are M samples all together,the Allan Variance is defined as[8] [20]:

$$\sigma_y^2(\tau) = \frac{1}{2}E\{y_{k+n} - y_k\} \tag{2.55}$$

Parameterizing the time interval $\tau = n\tau_0$ yields the fractional frequency:

$$y_k = \frac{x_{k+n} - x_k}{\tau} = \frac{x_{k+n} - x_k}{n\tau_0} \tag{2.56}$$

The Allan variance can be rewritten as[26]:

$$\sigma_y^2(\tau) = \frac{1}{2}E\{(x_{k+2n} - 2x_{k+n} + x_k)^2\} = \frac{1}{2\tau^2(M-2n)} \sum_{k=1}^{M-2n} (x_{k+2n} - 2x_{k+n} + x_k)^2 \tag{2.57}$$

This is most commonly used form, because it can be calculated directly from the time difference samples.

Allan variance depends on the time between samples and plotted in double logrithmos scale as a function of the time interval. Allan variance describes the stability of the clock whereas a lower Allan variance results on an increased stability.

2.6 System noise model of Kalman Filter

The system noise and measurement noise (Q and R) are both important parameters for Kalman filter. These 2 parameters directly effect the filtering effect. The measurement noise depends on the accuracy of the sensors and the measurement equipments. The system noise model in the "clock" case depends on the type of the noise source. For example: white noise, flicker noise, random walk noise [9]. There is no standard method of system noise modulation. There are some ways to avoid this parameter such as:Adaptive Kalman filter or manually input base on the experience and filter performance.

Adaptive Kalman filter is another interesing topic. It updates the error variance Q and R automatically during the operation. A more precise model of Q and R might result in a lower Allan variance. The system and measurement noise information is acquired from the measurement. The filtering residuals contain the information of the Q and R. References [7] [13] and [16] present several realizations of the adaptive Kalman filter. Because of the limit of time, in this diplom work the adaptive kalman filter is not implemented. Another way of acquiring system noise model is applied.

In reference [2] a method to determine the parameters of Q for a 2 states model is described.

The clock states are

$$X(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}$$

The system noise model is a covariance of the time and frequency error, and is defined as:

$$Q = \begin{bmatrix} q11 & q12 \\ q21 & q22 \end{bmatrix} = COV[x(t), y(t)] = \begin{bmatrix} \sigma_{x_0}^2 + \sigma_{x_{-1}}^2 + \sigma_{x_{-2}}^2 & R_{xy_0} + R_{xy_{-1}} + R_{xy_{-2}} \\ R_{xy_0} + R_{xy_{-1}} + R_{xy_{-2}} & \sigma_{y_0}^2 + \sigma_{y_{-1}}^2 + \sigma_{y_{-2}}^2 \end{bmatrix} \tag{2.58}$$

where The variances $\sigma_{x_0}^2$,$\sigma_{x_{-1}}^2$,$\sigma_{x_{-2}}^2$ describe the variances of different types of phase noise, and $\sigma_{y_0}^2$,$\sigma_{y_{-1}}^2$,$\sigma_{y_{-2}}^2$ describe the variances of different types of frequency noise.

The noise spectral density is converted to a single-sided spectral density of fractional frequency and modeled according to [4][14][17] as:

$$S_y(f) = h_2 f^2 + h_1 f + h_0 + h_{-1}/f + h_{-2}/f^2; \qquad f_l \leq f \leq f_h \tag{2.59}$$

where f_l - f_h defines the considered measurement system noise band width, and where the h_α coefficients represent the following processes:
h_2 - white phase noise
h_1 - flicker phase noise
h_0 - white frequency noise
h_{-1} - flicker frequency noise
h_{-2} - random walk frequency noise

These h's are associated with the Allan Variance [11]:

$S_y(f)$	$\sigma_y^2(\tau)$
$h_2 f^2$	$h_2 f_h/4\pi^2\tau^2$
$h_1 f$	$h_1[1.04 + 3ln(2\pi f_h)]/4\pi^2\tau^2$
h_0	$h_0/2\tau$
$h_{-1}f^{-1}$	$2h_{-1}ln(2)$
$h_{-2}f^{-2}$	$2\pi^2 h_{-2}\tau/3$

Table 2.1: Correspondence between the various noise components perturbing an oscillator and the Allan variance of the normalized frequency fluctuations

To get the description of other types of noise from the Allan variance, it is necessary to relate the other noise types to Gaussian white noise. The spectral density of the white noise is a constant, in this case h_0. The other spectral densities can be related to a white noise spectral density by:

$$S_Z(\omega) = |H(j\omega)|^2 S_W(w) \tag{2.60}$$

where $S_W(w)$ is the spectral density of the white noise,
$S_Z(w)$ is the spectral density of other types of noises.

Then the following spectral density S' are obtained:

$$S'_{y_0}(\omega) = h_0/2 \qquad \text{(white frequency noise)} \tag{2.61}$$

$$S'_{y_{-1}}(\omega) = \pi h_{-1}/\omega \qquad \text{(flicker frequency noise)} \tag{2.62}$$

$$S'_{y_{-2}}(\omega) = 2\pi^2 h_{-2}/\omega^2 \qquad \text{(random walk frequency noise)} \tag{2.63}$$

$$S'_{x_0}(\omega) = h_0/\omega^2 \tag{2.64}$$

$$S'_{x_{-1}}(\omega) = \pi h_{-1}/\omega^3 \tag{2.65}$$

$$S'_{x_{-2}}(\omega) = 2\pi^2 h_{-2}/2\omega^4 \tag{2.66}$$

The respective impulse responses are given by equation (2.60) and inverse Fourier transformation [2]:

$$h_{y_0}(t) = \sqrt{h_0/2} \cdot \delta(t) \tag{2.67}$$

$$h_{y_{-1}}(t) = \sqrt{h_{-1}/t}; t > 0 \tag{2.68}$$

$$h_{y_{-2}}(t) = \pi\sqrt{h_{-2}}; t \geq 0 \tag{2.69}$$

$$h_{x_0}(t) = \sqrt{h_0/2}; t \geq 0 \tag{2.70}$$

$$h_{x_{-1}}(t) = 2\sqrt{h_{-1}t}; t \geq 0 \tag{2.71}$$

$$h_{x_{-2}}(t) = \pi\sqrt{2h_{-2}t}; t \geq 0 \tag{2.72}$$

In the following section, the variances and covariances that contributes to the system noise covariance matrix are deduced. The variances are based on the autocorrelation function $R(t,\tau)$:

$$R(t,\tau) = \int_0^t \int_0^{t+\tau} h(t-u)h(t+\tau-v)E[\eta(u)\eta(v)]dvdu \tag{2.73}$$

Using the property that

$$E[\eta(u)\eta(v)] = \delta(u-v) \tag{2.74}$$

and

$$\int_0^{t+\tau} f(v)\delta(u-v) = f(u) \tag{2.75}$$

so that

$$R(t,\tau) = \int_0^t h(u)h(u+\tau)du; \tau \geq 0 \tag{2.76}$$

The autocorrelation function for each process are then [2]:

$$R_{y_0}(t,\tau) = \frac{h_0}{2}\delta(\tau) \tag{2.77}$$

$$\begin{aligned} R_{y_{-1}}(t,\tau) &= h_{-1}\int_{2\pi f_1}^{2\pi f_h} \frac{cos\omega\tau}{\omega}d\omega \\ &= h_{-1}ln\frac{f_1}{f_h} + h_{-1}\sum_{n=1}^{\infty}(-1)^n\frac{(2\pi f_h\tau)^{2n} - (2\pi f_1\tau)^{2n}}{2n(2n)!} \end{aligned} \tag{2.78}$$

$$R_{y_{-2}}(t,\tau) = 2\pi^2 h_{-2}t; \tau \geq 0 \tag{2.79}$$

$$R_{x_0}(t,\tau) = \frac{h_0}{2}t; \tau \geq 0 \tag{2.80}$$

$$R_{x_{-1}}(t,\tau) = h_{-1}\{(2t+\tau)\sqrt{t^2+t\tau} - \frac{\tau^2}{2}\ln[\frac{2t+\tau+2\sqrt{t^2+t\tau}}{\tau}]\} \tag{2.81}$$

$$R_{x_{-2}}(t,\tau) = 2\pi^2 h_{-2}(\frac{1}{3}t^3 + \frac{1}{2}t^2\tau) \tag{2.82}$$

The variance can be calculated from:

$$\sigma(t) = R(t,0) = \int_0^t h^2(u)du; \tag{2.83}$$

Using equation 2.83 the variances can be derived as:

$$\sigma_{y_0}^2 = h_0 f_{h_0} \tag{2.84}$$

$$\sigma_{y_{-1}}^2 = h_{-1} \ln \frac{f_h}{f_1} \tag{2.85}$$

$$\sigma_{y_{-2}}^2 = 2\pi^2 h_{-2} t \tag{2.86}$$

$$\sigma_{x_0}^2 = \frac{h_0}{2} t \tag{2.87}$$

$$\sigma_{x_{-1}}^2 = 2h_{-1}t^2 \tag{2.88}$$

$$\sigma_{x_{-2}}^2 = \frac{2}{3}\pi^2 h_{-2} t^3 \tag{2.89}$$

Similarly, the cross-correlation function between two processes is

$$R_{xy}(t,\tau) = \int_0^t h_x(u) h_y(u+\tau)du; \tau \geq 0 \tag{2.90}$$

The cross correlations between frequency and time are:

$$R_{xy_0} = \frac{h_0}{2} \tag{2.91}$$

$$R_{xy_{-1}} = 2h_{-1}t \tag{2.92}$$

$$R_{xy_{-2}} = \pi^2 h_{-2} t^2 \tag{2.93}$$

Combining equations (2.84)- (2.93) and (2.58), the covariance matrix of instantaneous time and frequency measurements is given by:

$$COV[x(t), y(t)] = \begin{bmatrix} \frac{h_0}{2}t + 2h_{-1}t^2 + \frac{2}{3}\pi^2 h_{-2}t^3 & \frac{h_0}{2} + 2h_{-1}t + \pi^2 h_{-2}t^2 \\ \frac{h_0}{2} + 2h_{-1}t + \pi^2 h_{-2}t^2 & h_0 f_{h_0} + h_{-1} ln\frac{f_h}{f_1} + 2\pi^2 h_{-2}t \end{bmatrix} \tag{2.94}$$

The fractional frequency is calculated from

$$y_t = \frac{x(t - \delta t) - x(t)}{\delta t} \tag{2.95}$$

The covariance matrix can be rewritten only with one variable t.

$$COV[x(\Delta t), \bar{y}(\Delta t)] = \begin{bmatrix} \frac{h_0}{2}\Delta t + 2h_{-1}\Delta t^2 + \frac{2}{3}\pi^2 h_{-2}\Delta t^3 & \frac{h_0}{2} + 2h_{-1}\Delta t + \pi^2 h_{-2}\Delta t^2 \\ \frac{h_0}{2} + 2h_{-1}\Delta t + \pi^2 h_{-2}\Delta t^2 & \frac{h_0}{2\Delta t} + 2h_{-1} + \frac{8}{3}\pi^2 h_{-2}\Delta t \end{bmatrix} \tag{2.96}$$

Replacing Δt with τ yield:

$$Q = \begin{bmatrix} \frac{h_0}{2}\tau + 2h_{-1}\tau^2 + \frac{2}{3}\pi^2 h_{-2}\tau^3 & \frac{h_0}{2} + 2h_{-1}\tau + \pi^2 h_{-2}\tau^2 \\ \frac{h_0}{2} + 2h_{-1}\tau + \pi^2 h_{-2}\tau^2 & \frac{h_0}{2\tau} + 2h_{-1} + \frac{8}{3}\pi^2 h_{-2}\tau \end{bmatrix} \tag{2.97}$$

Finally the system noise matrix can be related to the parameters h_i, $i\epsilon\{-2, \ldots, +2\}$which can be computed from the Allan variance.

3 Applying Kalman Filter in different cases

The most important part of the clock steering system is the Kalman filter. It's important to test all the features related to the implementation. In this chapter, Kalman filter applied in position/speed case and clock cases with different data rates are tested.

3.1 Test Kalman Filter

3.1.1 Position/Speed Case

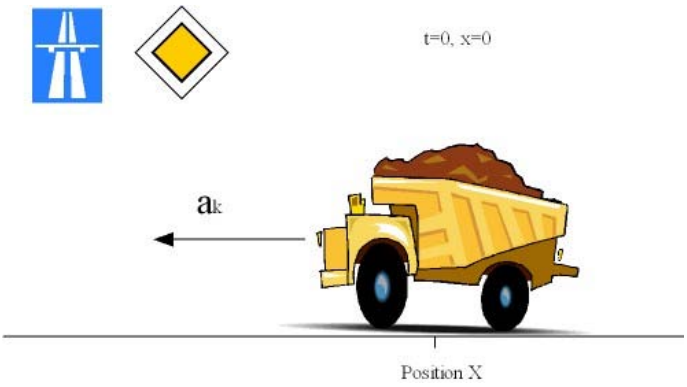

Figure 3.1: Position/Speed Case

To see how the Kalman filter works, a simple application is chosen. It is supposed that a vehicle moves on the infinitely long, straight, frictionless road. Initially it is stationary at the position 0, but from t=0 on it is accelerated by a random acceleration. The position of the vehicle is observed every Δt seconds. [22]

The state vector of the system is :

$X_k = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$,

where x is the position and $\dot{x}$ is the velocity at k_{th} observation.

The acceleration a_k is generated by the random function. It's normally distributed with zero mean and standard deviation σ_a.

From Newton's laws of motion, it is concluded that

$$x_k = x_{k-1} + \dot{x}_{k-1}\Delta t + \frac{\Delta t^2}{2} a_k \tag{3.1}$$

$$\dot{x}_k = \dot{x}_{k-1} + \Delta t a_k \tag{3.2}$$

Define

$$F = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{3.3}$$

and

$$G = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} \tag{3.4}$$

The system equation can be written as:

$$X_k = F X_{k-1} + G a_k \tag{3.5}$$

The system noise Q is:

$$Q = cov(Ga) = E[(Ga)(Ga)^T] = GE[a^2]G^T = G[\sigma_a^2]G^T = \sigma_a^2 G G^T \tag{3.6}$$

With these assumptions the noised measurement can be rewritten as:

$$Z_k = H \cdot X_k + V_k \tag{3.7}$$

where H is the observation model and $H = \begin{bmatrix} 1 & 0 \end{bmatrix}$,

v_k is the measurement noise, the measurement noise variance for Kalman filter $R = E[v_k v_k^T] = [\sigma_z^2]$

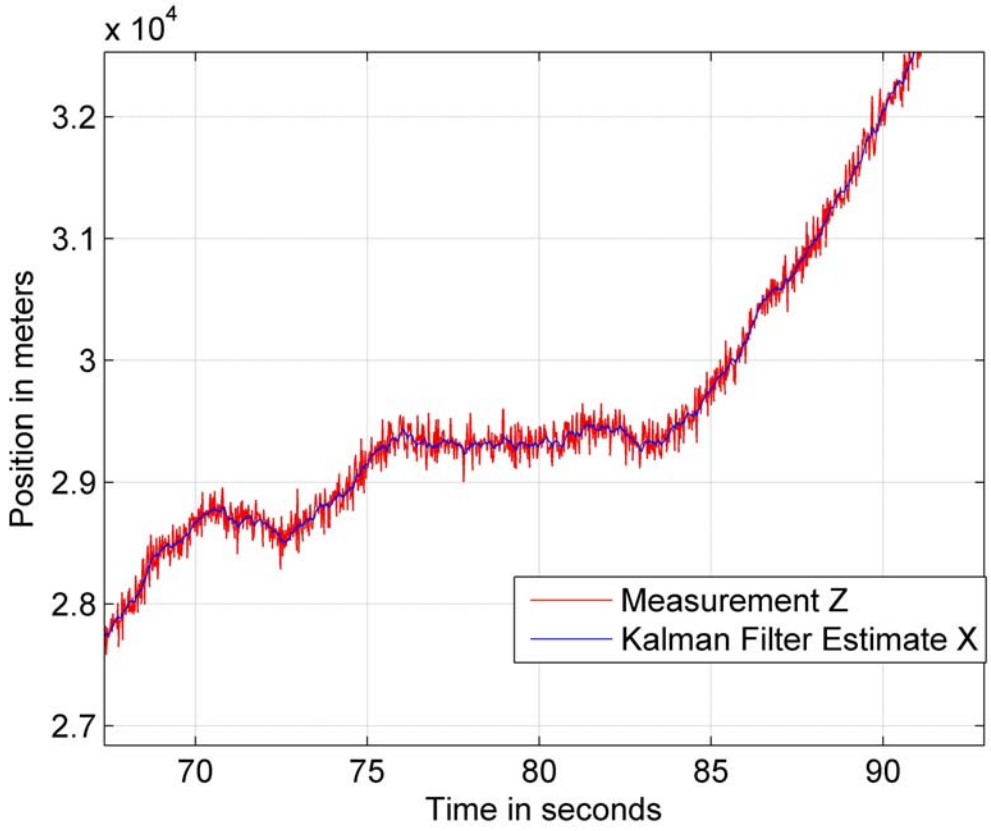

Figure 3.2: Position estimation with Kalman filter

Using the standard equation of the Kalman filter, after the simulation the result is shown in the following figure.

From the figure it can be seen that the Kalman filter filtered out the system noise and measurement noise. It gives smoothed output.

The mean square error(MSE) can be calculated with the equation:

$$MSE(X) = E((X - \theta)^2) \tag{3.8}$$

where θ is the true value.

The mean square error of the measurement $MSE_{Measurement}$= 1.0071×10^4
The mean square error of the Estimation $MSE_{Estimation}$= 160.0138

3.1.2 Time Difference Estimation with Simulated Data

Herein the kalman filter is used to steer a clock system. After the first study of the position-speed case, the clock system is bulit with the following system equation[15]:

$$X_{k+1} = \phi X_k + W_k \tag{3.9}$$

$$Z_k = H \cdot X_k + V_k \tag{3.10}$$

where ϕ is the transition matrix and defined as : $\begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix}$, τ is the time interval between measurements.

$W(k)$ is Gaussian white noise characterized by covariance.

$X(k)$ is the state vector, defined as: $\begin{bmatrix} x_1(k) \\ y_2(k) \end{bmatrix}$,

where $x_1(k)$ is the time difference, and $y_2(k)$ is the fractional frequency difference, between the reference and the steered standard.

H is the connection matrix and $H := [1\ 0]$, and v_k is the Measurement noise.

With a set of simulated data containing the time difference between the clock which need to be steered and the reference clock the Kalman filter was tested again.

In this test the measurement noise with standard deviation σ=3ns(white noise) was added to the simulated data. This test is just to make sure that the Kalman filter works properly with this clock model and the filter converges. The Figure 3.3 shows that Kalman filter filtered out the measurement noise. Because the data fixed in file and there is no interactively changing to the data, the time difference can just be estimated but not be reduced.

The mean square error(MSE) of the measurement $MSE_{Measurement}$= 9.1047e-018
The mean square error of the Estimation $MSE_{Estimation}$= 2.7090e-019

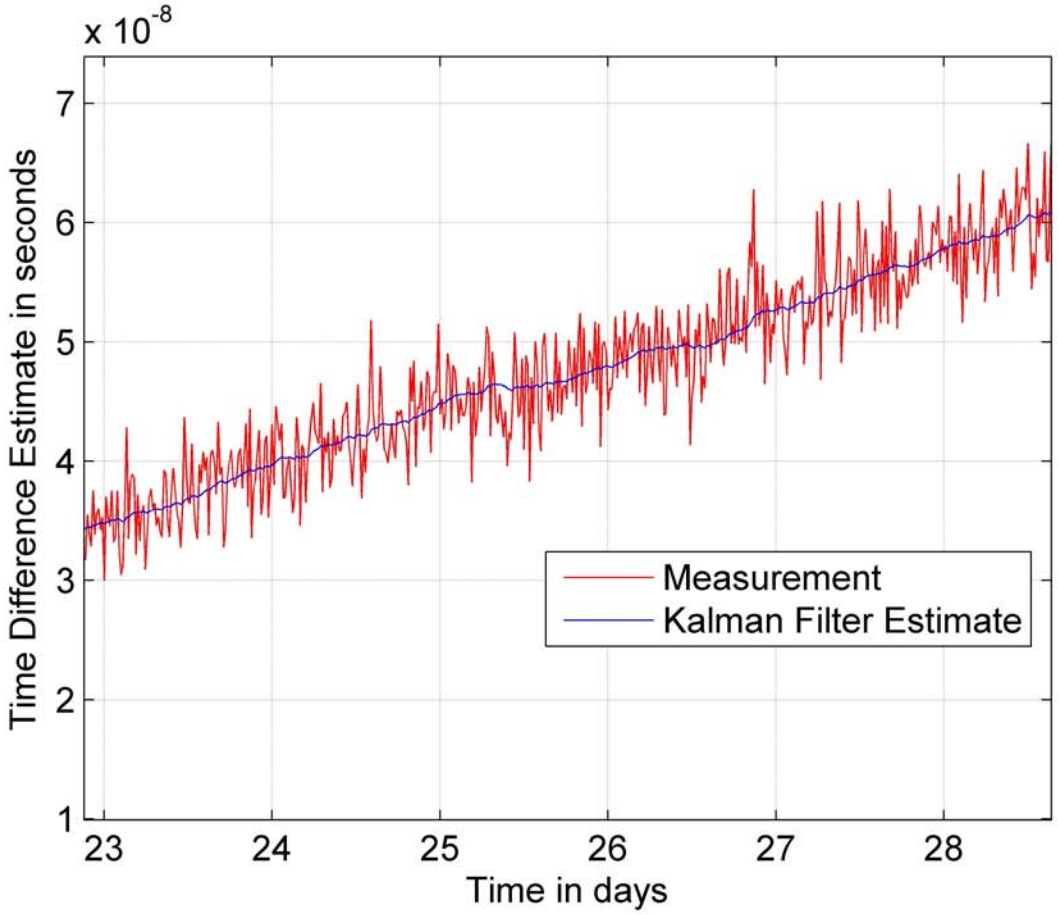

Figure 3.3: Time Estimate with Simulated data

3.1.3 Time Difference Estimation with interactively generated Measurements

The filtering function of the steering system has been tested by the last test. It's important to know if the system can keep the time difference around zero. To achieve this goal, the measurements has to be generated dynamically according to the steering of the earlier steps. And the controlling effect has also to be considered inside the Kalman filter.

The equation to dynamically generating the measurements is:

$$z_{k+n} = z_{k+n} + n \cdot \tau u_k \tag{3.11}$$

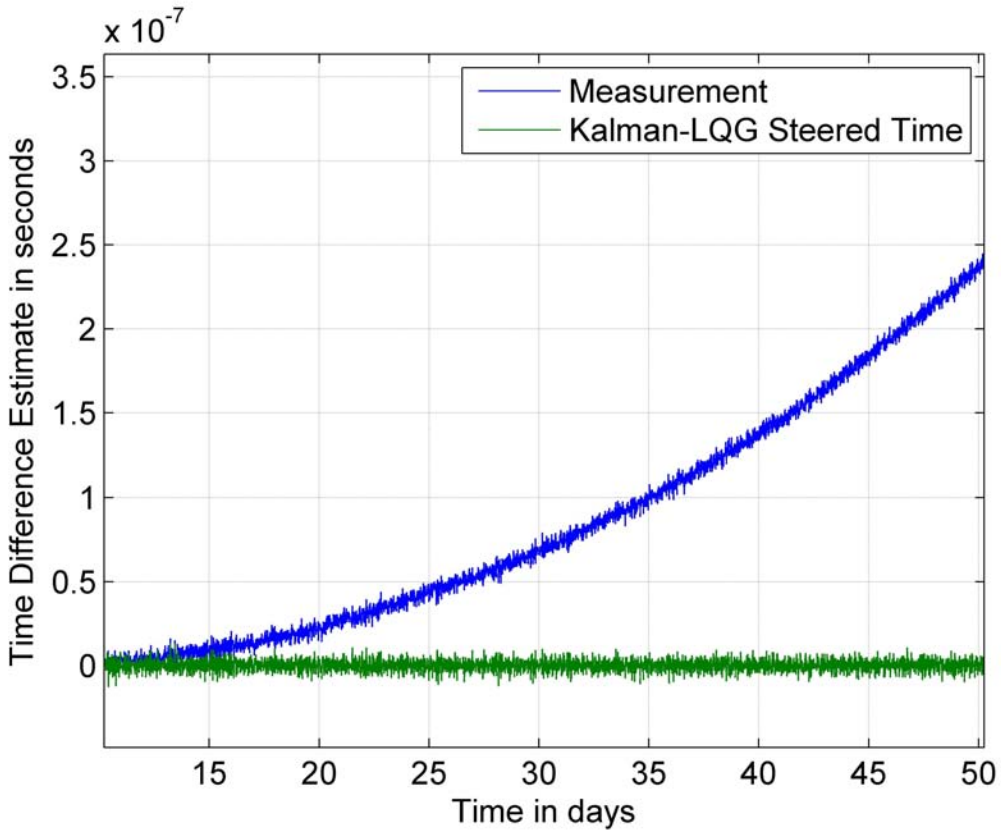

Figure 3.4: Time Difference Estimate with interactive Measurements

The equation of one-step-ahead prediction has to be changed to:

$$X_{k+1} = \phi X_k + Bu_k + W_k \tag{3.12}$$

Figure 3.4 shows the comparison of the measurements and the Kalman-LQG steered time. It can be seen that the system can keep the time difference around zero. The time difference will keep increasing as the case described in section 3.1.2 if there is no steering.

The standard deviations are:

$$\sigma_{Measurement}=2.5836\times10^{-7}$$
$$\sigma_{Kalman-LQG}=3.4990\times10^{-9}$$

3.1.4 Time Difference Estimate with interactive Measurements and offsets

After finishing the test in section 3.1.3, it's known that the implemented Kalman filter can keep the time around the zero value. The following test is to see if the system works properly if a very big offset suddenly appears and if the system can drive the time difference back to zero in that case. In this test 2 offsets of 178.51 nanoseconds were added to the measurements.

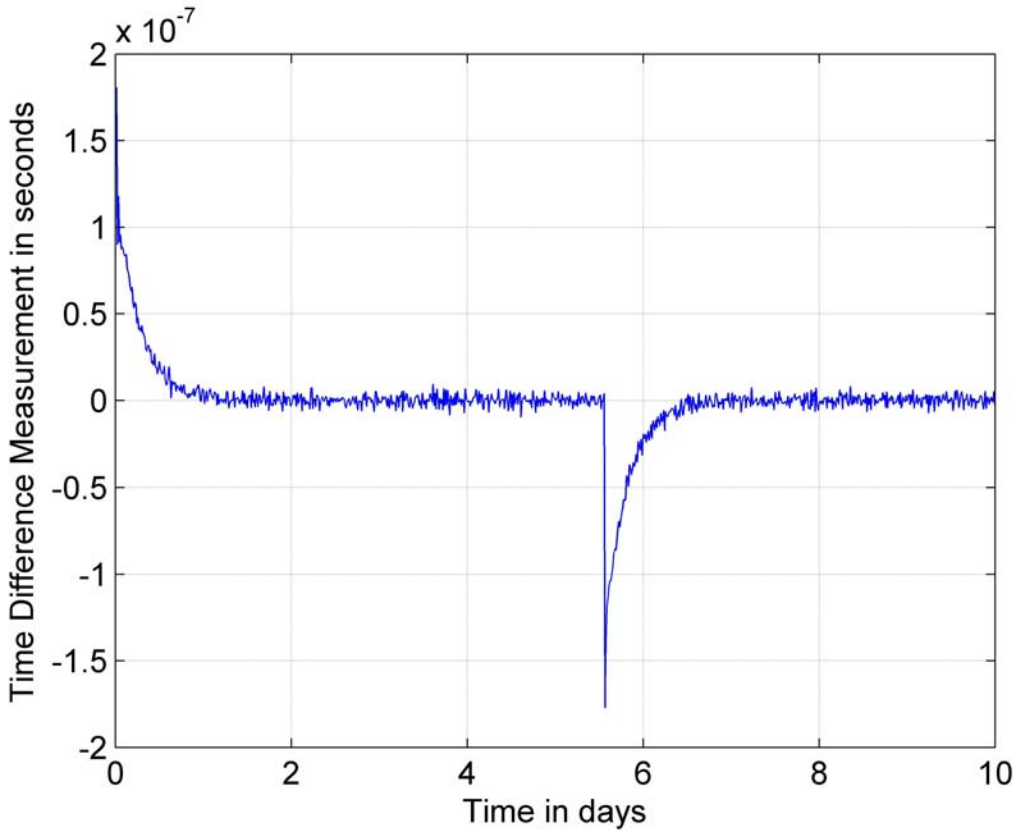

Figure 3.5: Time Difference Estimate with interactive Measurements and offsets

Figure 3.5 shows that the system can also steer the time difference towards zero even a very big offset was suddenly added to the system.The speed at which the system steer the time difference to zero can be adjusted by modifying the parameters WQ and WR in the LQG regulator.

3.1.5 Time Difference Estimate with Measurements from 2 days ago

The simulated data are generated in 16 minutes interval. But in real implementation the data can only be acquired 2 days after the offset happened. The Kalman filter is changed to work at the interval of 2 days and the control elements are calculated from the one step ahead prediction. 2 stages of Kalman filter are applied in this case.

Assume the current time is t, the steering system with 2 stages of Kalman filter can be described with the following figure:

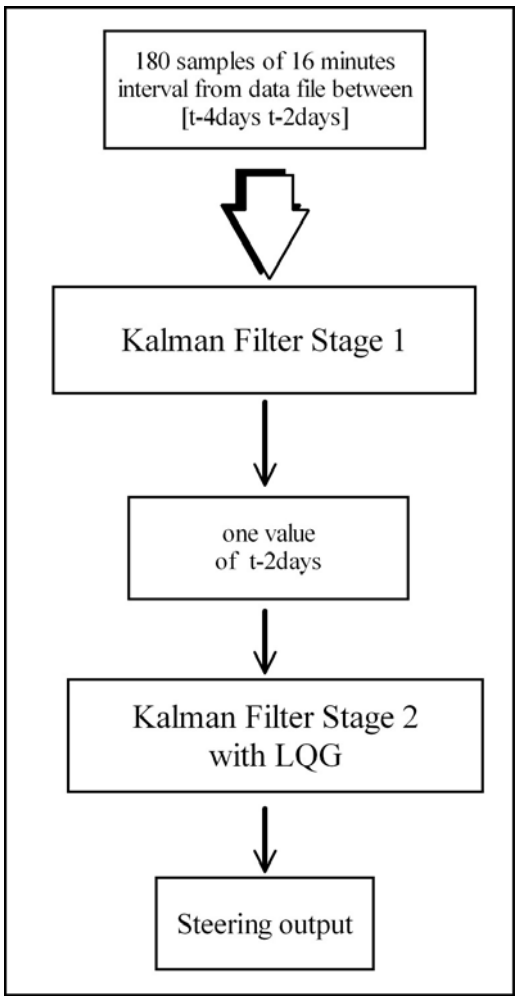

Figure 3.6: Steering system 2 stages of Kalman Filter

With 2 days interval, all the last 3 tests are repeated. One of the most different part

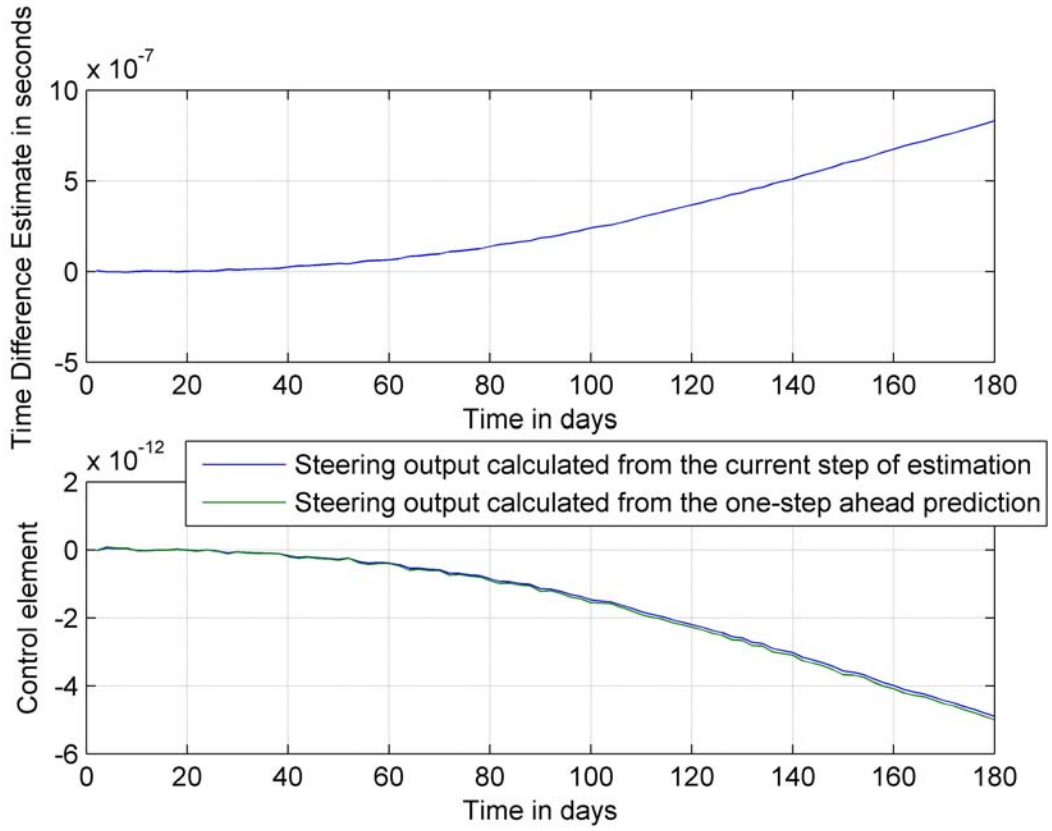

Figure 3.7: Time Difference Estimate with Measurements of 2 days interval

between 16 minutes interval and 2 days interval is: The steering output is no longer calculated from the time difference estimate, but from the one-step-ahead prediction.

$$u(k) = -\hat{G}_0 \hat{x}_{k+1/k}(k) \tag{3.13}$$

The figure shows that the Kalman filter works also properly with 2 days interval. The Kalman filter keep converge. Figure 3.7 shows that the steering output calculated from the one-step-ahead prediction is similar to which calculated from the current step of estimation.

3.1.6 Time Difference Estimate with interactively generated Measurements from 2 days ago without offsets

It's important to keep the Kalman filter converge. The noise variance Q, R and WR, WQ for LQG regulator can all cause the Kalman diverging. It's very necessary test all possible situations. This trail is to test if the system works properly by taking steering output in consideration.

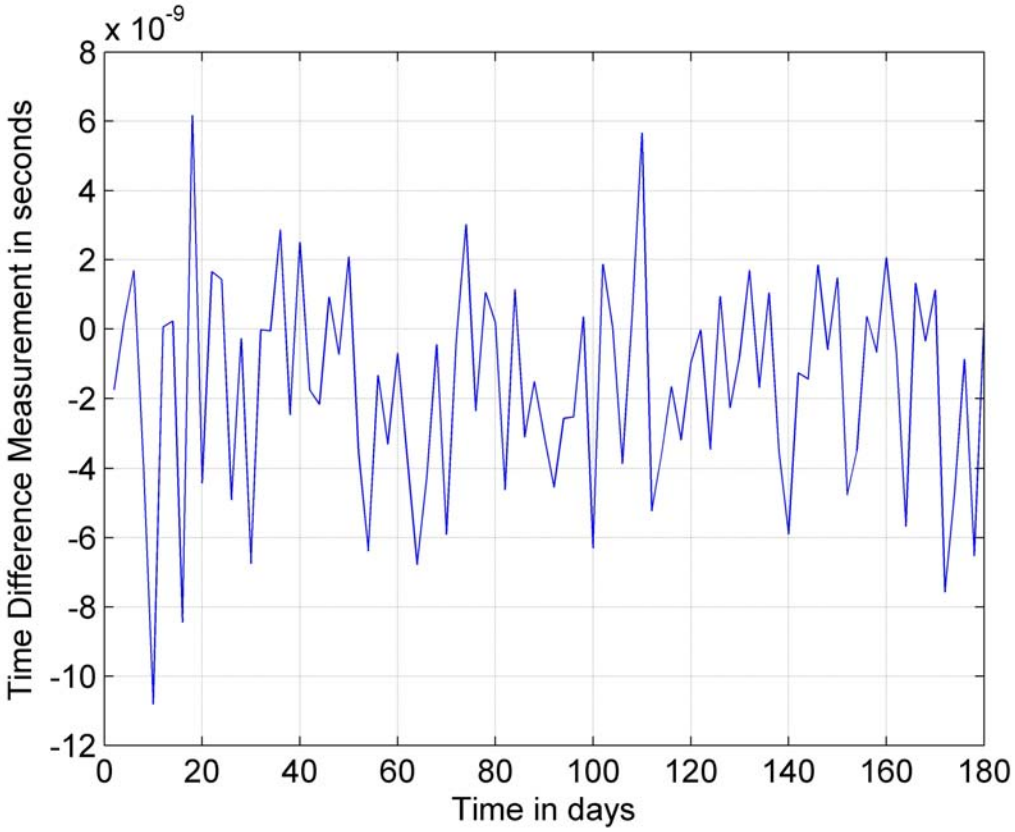

Figure 3.8: Time Difference Estimate with interactively generated Measurements from 2 days ago without offsets

This test is similar with the 16 minutes interval test(3.1.3). The system can also keep the time difference around the X-axis. This proves at least that the system doesn't have the parameters error. The system uses one-step-ahead prediction generated steering output can also keep the time difference near zero.

The standard deviation of the steered time difference: $\sigma = 2.6773 \times 10^{-9}$

3.1.7 Time Difference Estimate with interactively generated Measurements from 2 days ago with Offsets

Then the interactively generated measurements with 2 offsets (178.51 nanoseconds) is tested. This test is the last theoretical test before the system goes online.

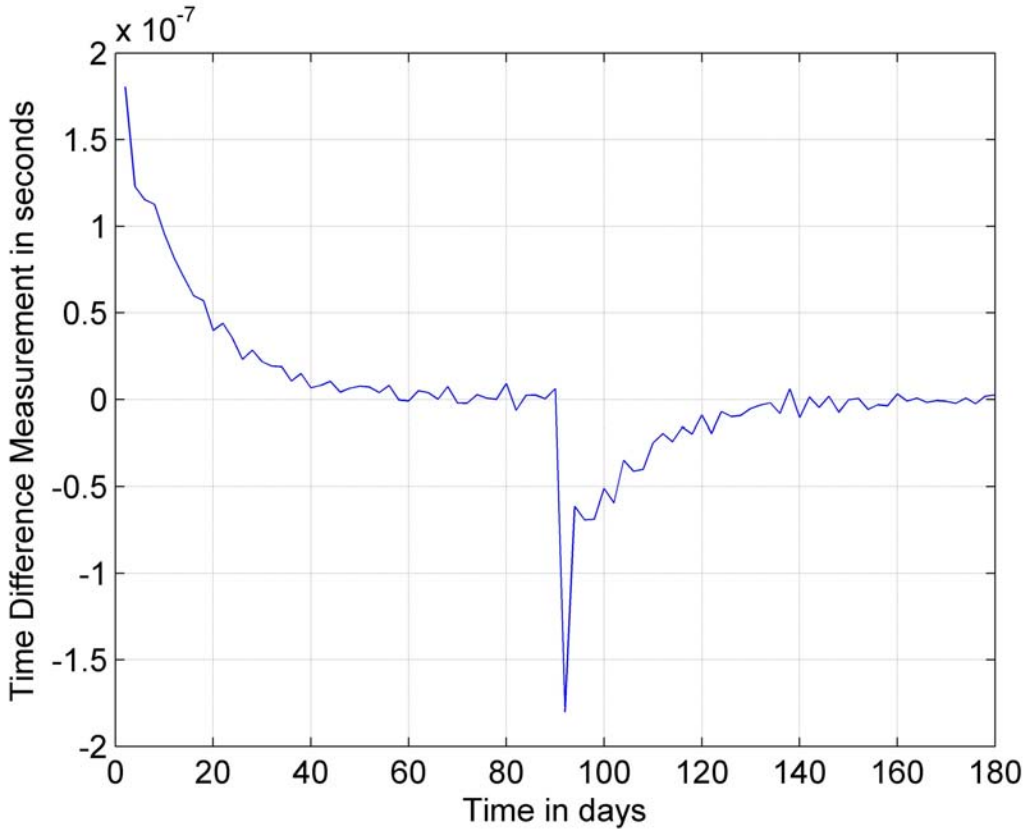

Figure 3.9: Time Difference Estimate with interactively generated Measurements from 2 days ago

The Kalman-LQG clock steering system is now implemented with 2 days interval. It calculates the steering output from the one-step-ahead prediction, takes steering output in consideration which can also be called close-loop operation.

The figure shows that the implemented filter make the filtering converge. It can keep the time difference around zero. It can also drive the big offset to zeros again. All the requirements for the implementation are meet.

3.2 Compare Kalman filter and Moving average

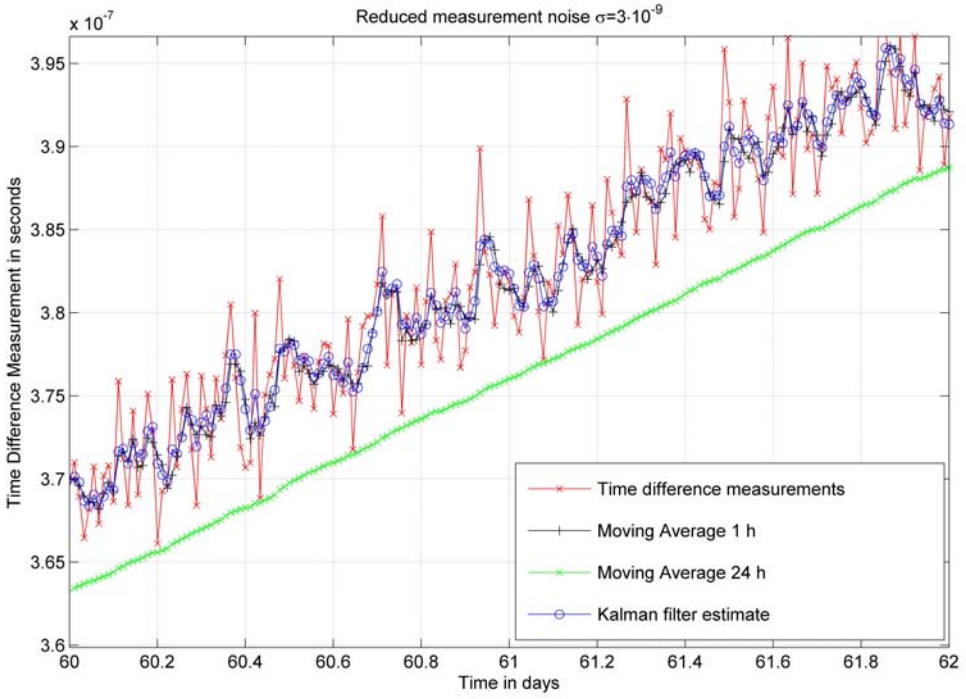

Figure 3.10: Compare Kalman filter and Moving average

Moving average is also an effective way of filtering out the noise. And it's easy to realize. So it's interesting to compare moving average with Kalman filter.

From Figure 3.10 it can be seen that the windows size of the moving average smoother can greatly affect the smoothing result. The 1 hour moving average has the similar effect as the Kalman filter. The 24 hours moving average looks much smoother than the others. This window size filtered out all the short term fluctuations.

Figure 3.11 shows that the moving average is a real "filter". Some useful information are filtered out.

	Mean square error
Measurement	2.7×10^{-17}
Moving average with 1 hour window size	3.8×10^{-18}
Moving average with 24 hour window size	3.6×10^{-17}
Kalman filter estimation	4.0×10^{-18}

Table 3.1: Compare the mean square errors of Figure 3.10

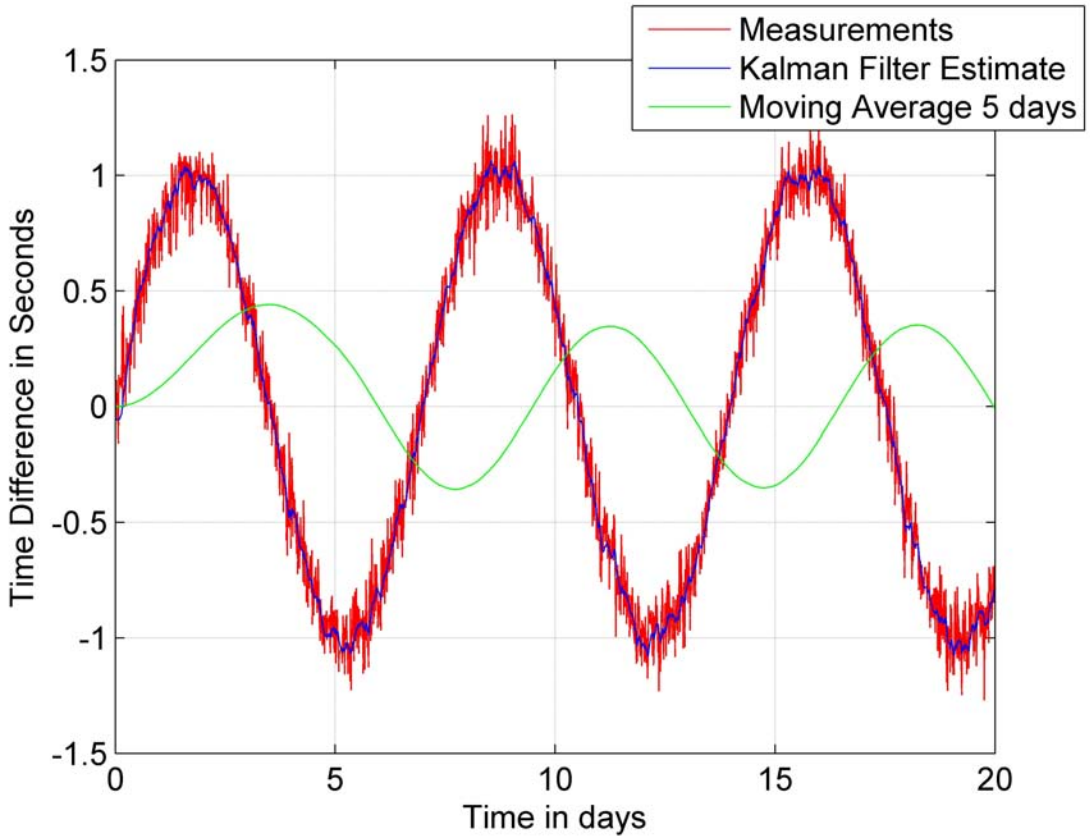

Figure 3.11: Special case of Moving average

The disadvantages of the moving average are:

1. Always consider about the historical data.
2. Lose some detail information about the measurement.

But if the noise is only Gaussian white noise and the window size is chosen properly, it

	Mean square error
Measurement	0.01
Moving average	0.7217
Kalman filter estimation	0.0012

Table 3.2: Compare the mean square errors of Figure 3.11

is still a good and simple way of filtering. In comparison, the Kalman filter is relative adaptive filter, it's no need to define window size for specified case.

Because moving average doesn't contain the system noise level information, and if the window size is not chosen properly, some useful information might be filtered out. Kalman filter contains the noise level information. It can filter out the system and measurement noise, and at the same time keep the useful information.After the comparison, the result is: Kalman filter is better choice than moving average.

4 Compare INPL Steering and Kalman-LQG Steering

Kalman-LQG is not the only method for clock steering. In this chapter, INPL method and Kalman-LQG method are compared. In first section the two methods are compared with 16 minutes data rate. In the second section they are compared with 2 days data rate. Because in the real implementation, the data is only available 2 days later after the offset take place.

The data file used in this chapter is a simulated data containing 21600 samples in 16 minutes step with the standard deviation of 3.7083×10^{-8}.

In the first section, the data rate is 16 minutes. There is no measurement noise added. In the second section, the measurement noise with standard deviation σ= 1 ns(white noise) is added to the simulated data.

4.1 Comparison with simulated data of 16 minutes interval

4.1.1 Steering with INPL method

INPL method is an rather simple method. Its advantage is the easy implementation. It's no need to input the system noise parameters or measurement noise parameter. In Figure 4.1, the offset and the frequency of the clock is calculated and steered every 16 minutes. It could obviously minimize the clock offset. But the curve have the form of scintillation. The steering with the INPL method is based on the historical measurement, there is no prediction for the future. Because of it, it is difficult to offer precise steering.

The standard deviation of the steered time offset: σ_{INPL}= 2.6117×10^{-9}

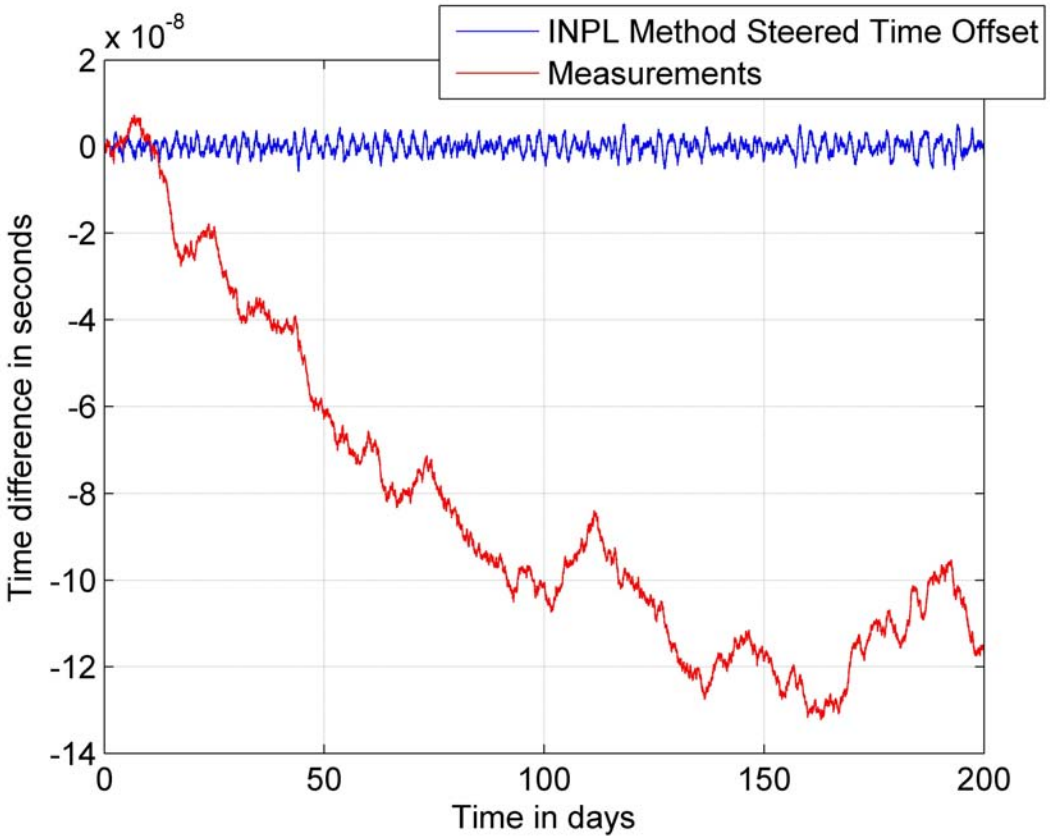

Figure 4.1: Steered time and time difference measurements

Equation (2.51) indicates that the steering policy of the INPL method is determined by the parameter m and l. To get the optimal steering 1600 combinations of these two parameters are tested, and the resulted Allan deviation are shown in Figure 4.2. From the figure it can be seen that most values in this case which make the steering converge have the similar Allan deviation. In this chapter, in all tests with INPL method, m=0.2 and $l = 0.05$ are chosen.

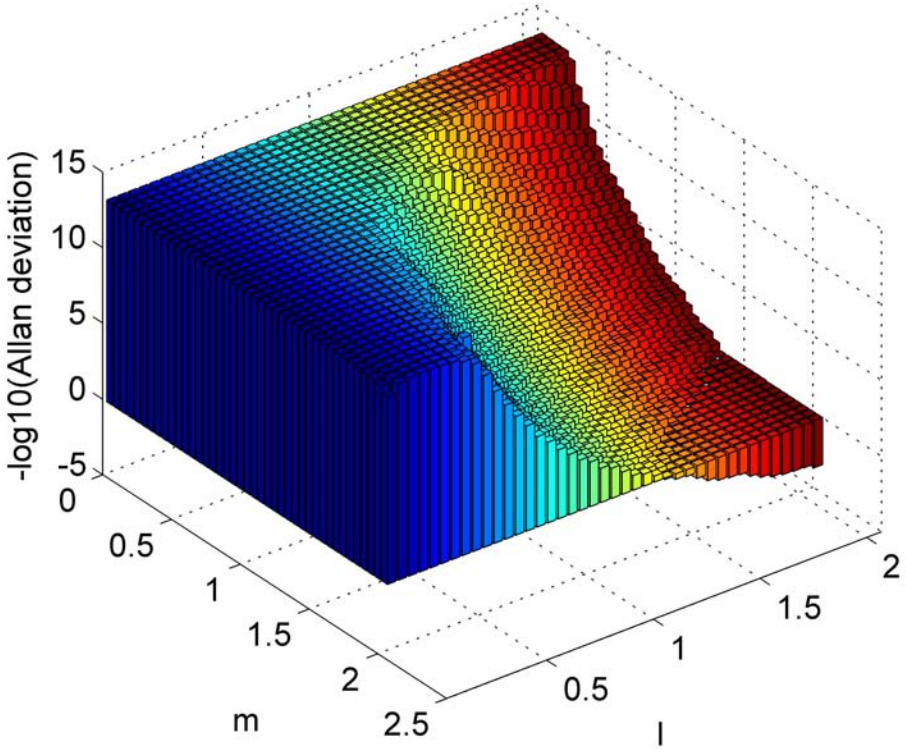

Figure 4.2: Allan deviation with different combinations of m and l

4.1.2 Steering with Kalman filter

Kalman-LQG method cares about both the accuracy and stability. The steered time offset in the figure looks much smoother. Because the system noise model and measurement noise model are input to the Kalman filter, the Kalman filter is able to estimates the time difference in the next step. The Kalman-LQG method offers more precise steering.

The standard deviation of the steered time offset: $\sigma_{Kalman-LQG}$= 4.0668×10^{-10}

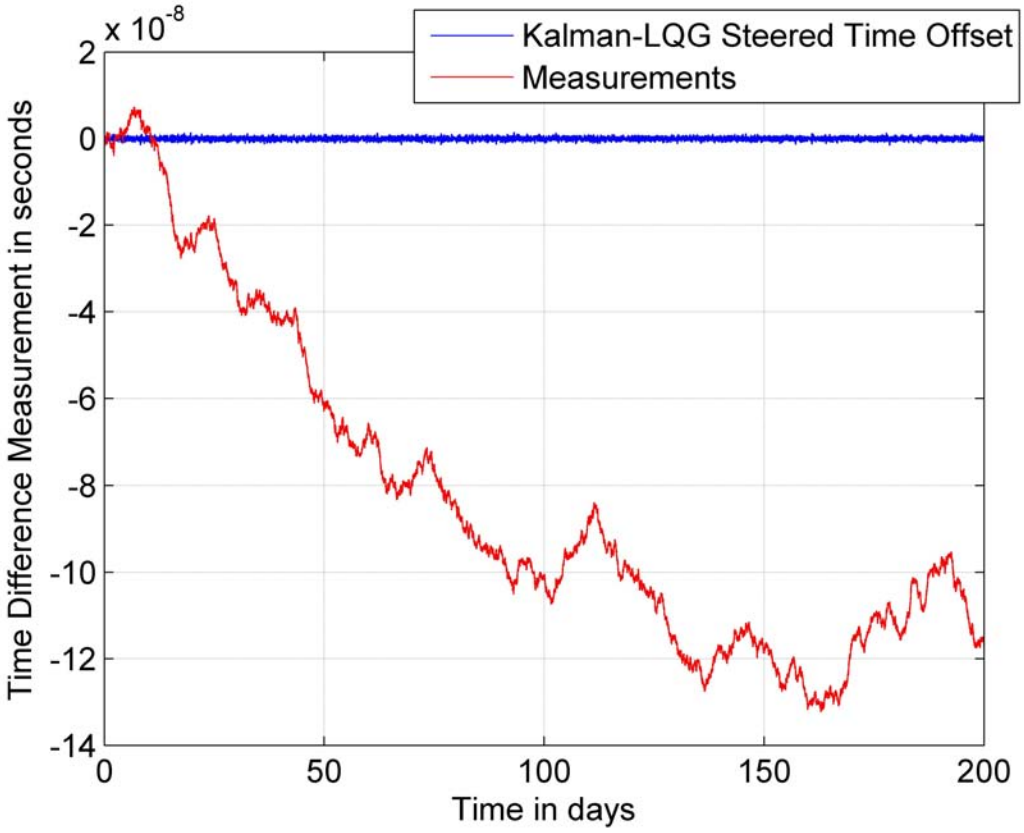

Figure 4.3: Kalman Steering

4.1.3 Compare the steered time

By the 16 minutes rate of steering, the Kalman-LQG steered time offset shows much better accuracy. The steering is very precise. The clock doesn't have to over strip first and then slow down like INPL method steered time. Any way, serval nanoseconds residual offset is a pretty good performance for both Kalman-LQG and INPL steered time.

Interval	INPL Method	Kalman-LQG Method
Measurement	$\sigma_{Measurement}$= 3.7803×10^{-8}	
16 Minutes	σ_{INPL}= 2.6117×10^{-9}	$\sigma_{Kalman-LQG}$= 4.0668×10^{-10}

Table 4.1: Standard deviations of INPL and Kalman-LQG methods steered time in 16 minutes interval

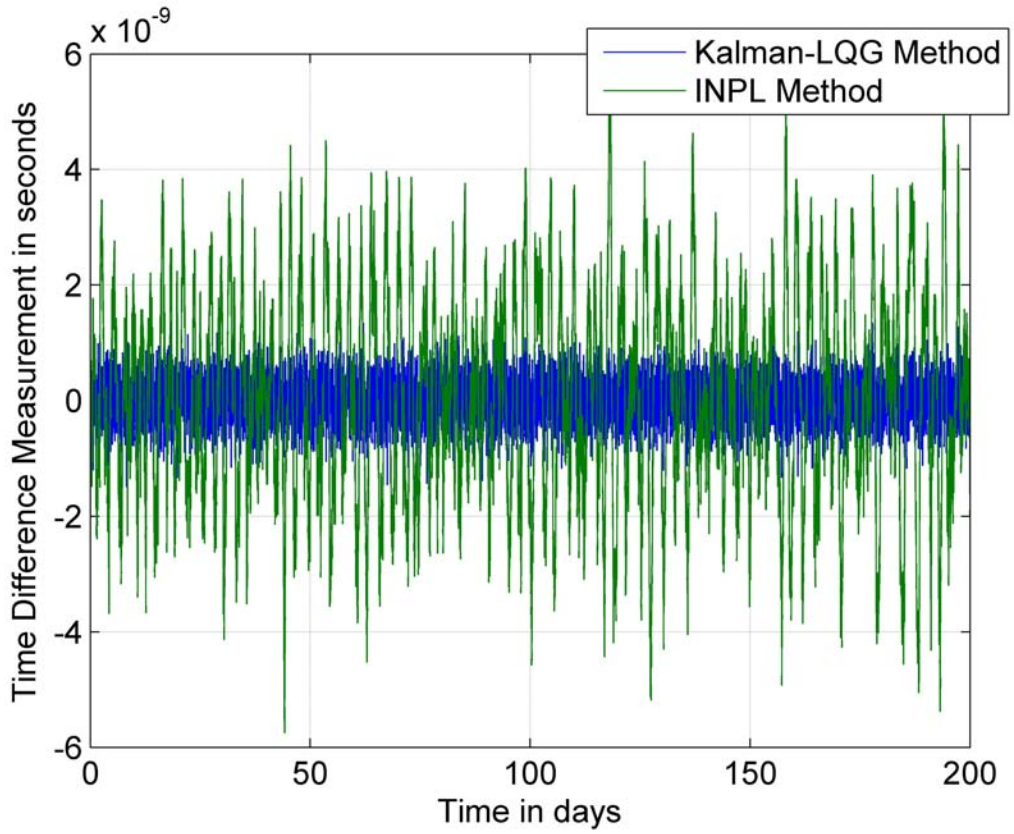

Figure 4.4: Comparison of INPL Steered time, Kalman-LQG Steered time offset and Original Measurements

4.1.4 Compare the Allan deviation

By comparing the two steering methods, it is shown in the figure that time steered by the Kalman-LQG method has both better accuracy and frequency stability(Figure 4.5). The Kalman-LQG steering has much better stability than the INPL-method and the measurements both in short term and long term.

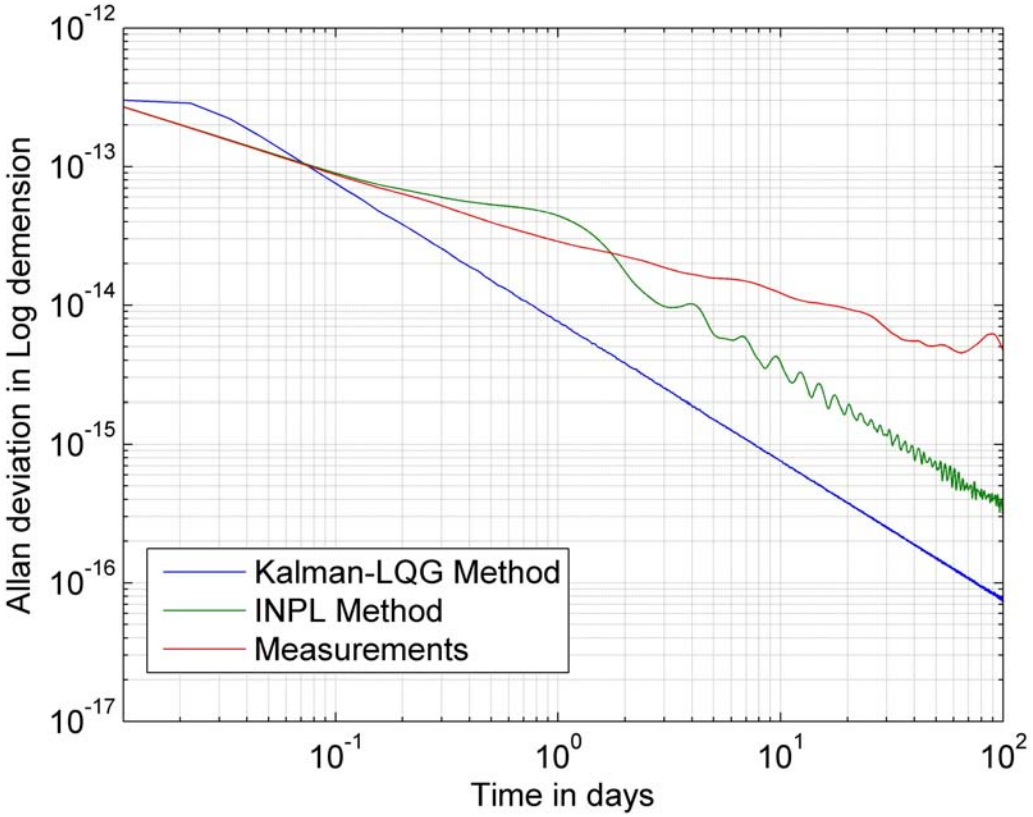

Figure 4.5: Comparison of Allan deviation

4.2 Comparison with simulated data of 2 days interval

In the current implementation, the data have 48 hours delay. It's also necessary to compare the INPL and Kalman-LQG method with 2 days data rate.

4.2.1 Steering with INPL method

To change INPL method to steer the time in 2 days interval, it's only needed to change the time interval value of the programme. Compare the Figure 4.1 and Figure 4.6, 2 days delay reduced the accuracy of the steered time offset.

The standard deviation of the steered time: σ_{INPL}= 4.4447×10^{-9}

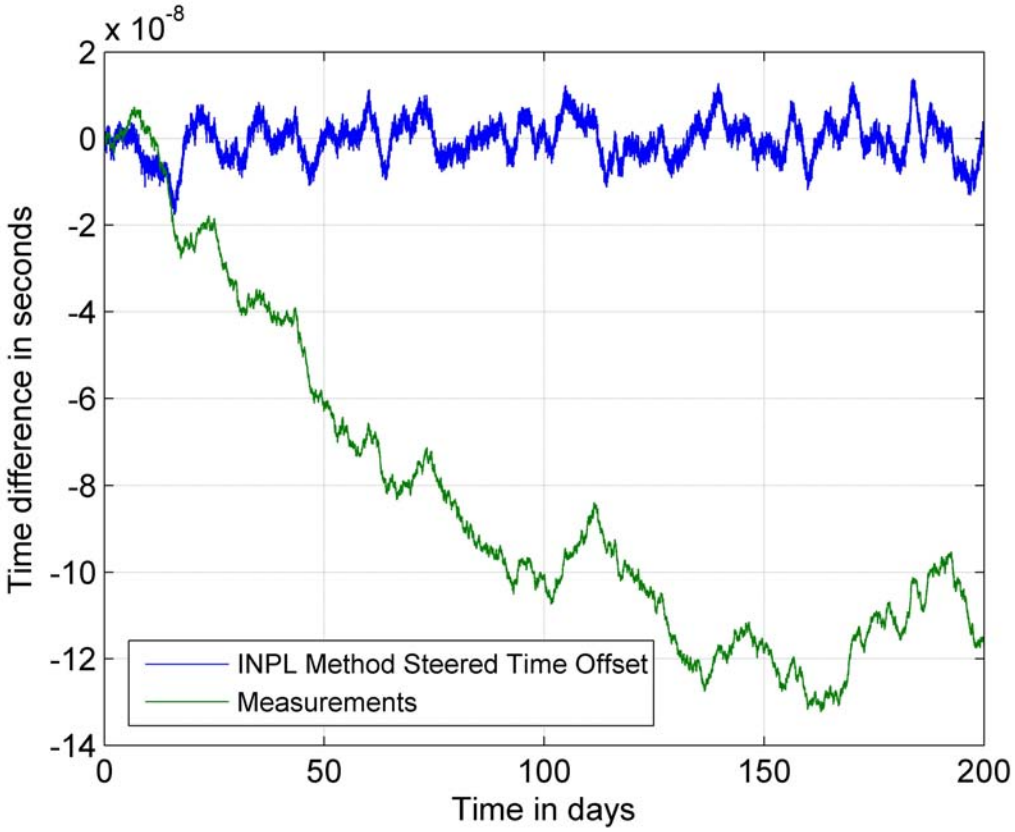

Figure 4.6: Steering with INPL method

4.2.2 Steering with Kalman filter

In 2 days case, 2 stages of Kalman filter [Described in Chapter 3] is applied. And one-step-ahead prediction is used to calculate the steering output. From the figure the time seems much more noisy than it was with the 16 minutes data interval. In this case, the Kalman-LQG steered time doesn't have sufficient steering rate. It also has to over strip and slow down all the time. The performance is no longer as good as it was in 16 minutes interval.

The standard deviation of the steered time: $\sigma_{Kalman-LQG}$= 7.5595×10^{-9}

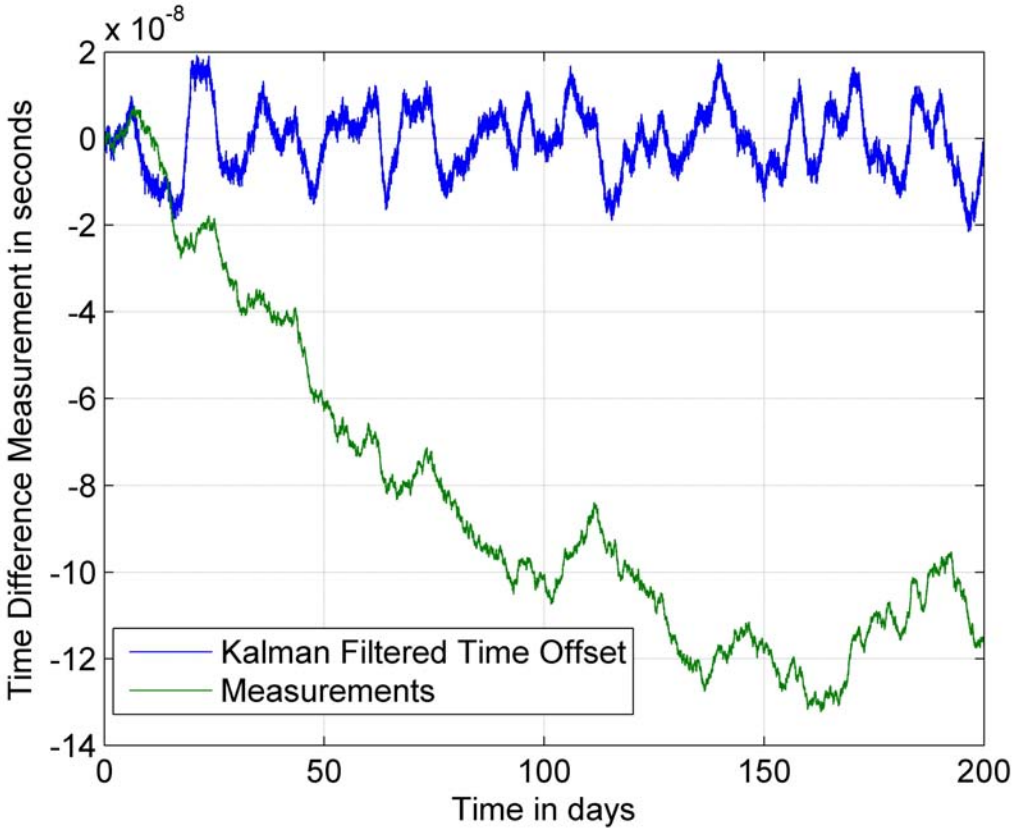

Figure 4.7: Steering with Kalman filter

4.2.3 Compare the steered time

If put them together, it can be seen that there isn't too much difference between the two methods in the current lab situation. Both curves have the form of scintillation. The INPL method steered time has smaller variance. By adjusting the parameters for both methods, the performance could be optimized. It can be expected that the Kalman-LQG method could be better in some cases. But the difference won't be too much. The performance of the two methods is similar in this case and stay at the same level. In this simulation the accuracy stays in 10 nanoseconds level.

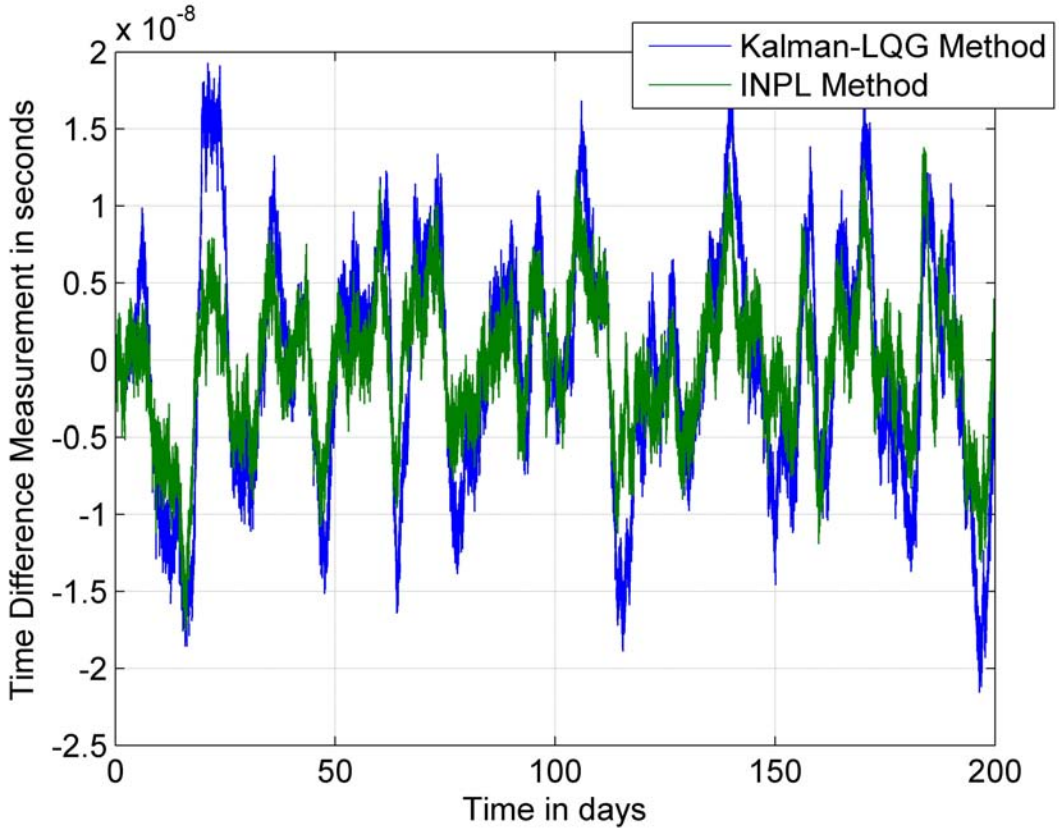

Figure 4.8: Compare the steered time offset

Interval	INPL Method	Kalman-LQG Method
Measurement	$\sigma_{Measurement}$= 3.7803×10^{-8}	
2 Days	σ_{INPL}= 4.4447×10^{-9}	$\sigma_{Kalman-LQG}$= 7.5595×10^{-9}

Table 4.2: Standard deviations of INPL and Kalman-LQG methods steered time in 2 days interval

4.3 Compare the steering effect with 16 minutes and 2 days data interval

With 16 minutes data interval, Kalman-LQG method is obviously better than the INPL one both in accuracy and stability. However, in the real operation conditions, the data has 48 hours delay and one-step-ahead prediction is applied. It's difficult to tell which method

is better in this situation. But in the implementation, Kalman-LQG method is still used, because the situation would become better in the future. The delay will become smaller and smaller. This means the Kalman-LQG steered clocks would then perform better. But the performance of the INPL method doesn't have too much space to improve.

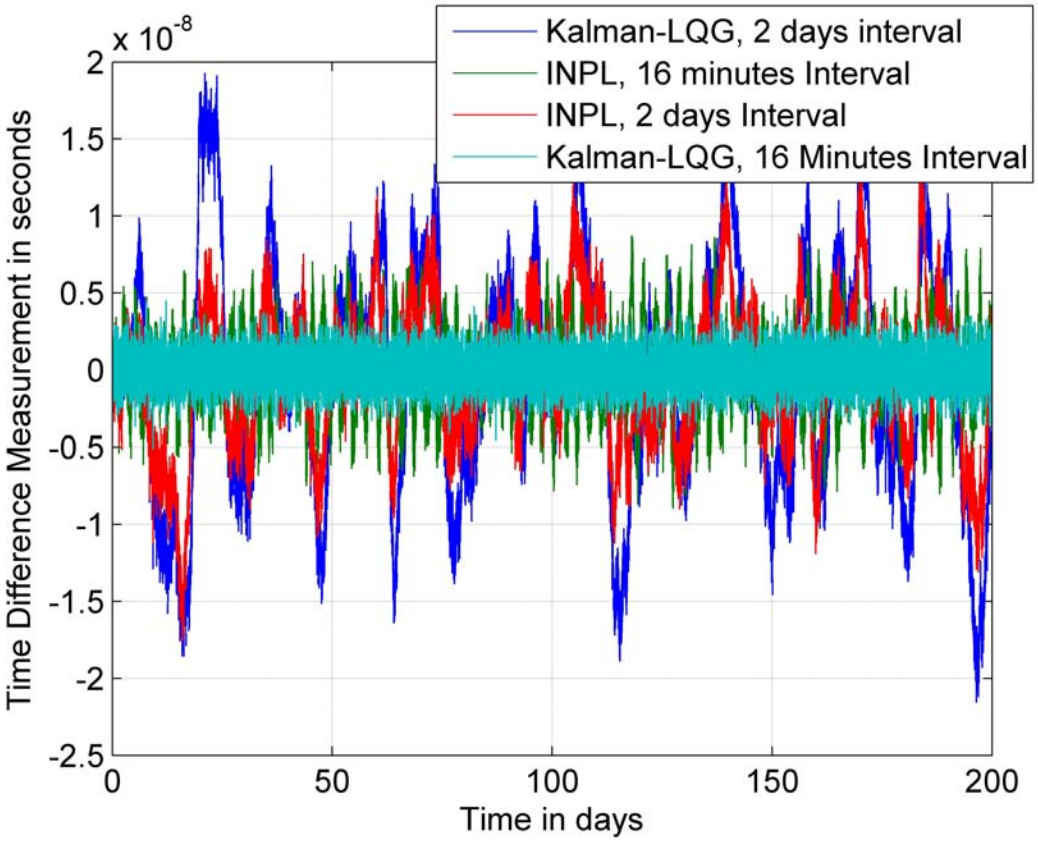

Figure 4.9: Compare the Allan deviation

Interval	INPL Method	Kalman-LQG Method
Measurement	$\sigma_{Measurement}$= 3.7803×10^{-8}	
16 Minutes	σ_{INPL}= 2.6117×10^{-9}	$\sigma_{Kalman-LQG}$= 4.0668×10^{-10}
2 Days	σ_{INPL}= 4.4447×10^{-9}	$\sigma_{Kalman-LQG}$= 7.5595×10^{-9}

Table 4.3: Standard deviations of INPL and Kalman-LQG methods steered time offset in different intervals

4.4 Steering effect with different data rates and delays

4.4.1 Kalman-LQG steered time with different data rates

It's interesting to know why in the the case of 2 days data rate case INPL method offers better accuracy than Kalman-LQG method. The data rate and delay must play a role inside. To prove this, both Kalman filter and INPL feeded with different data rates are tested.

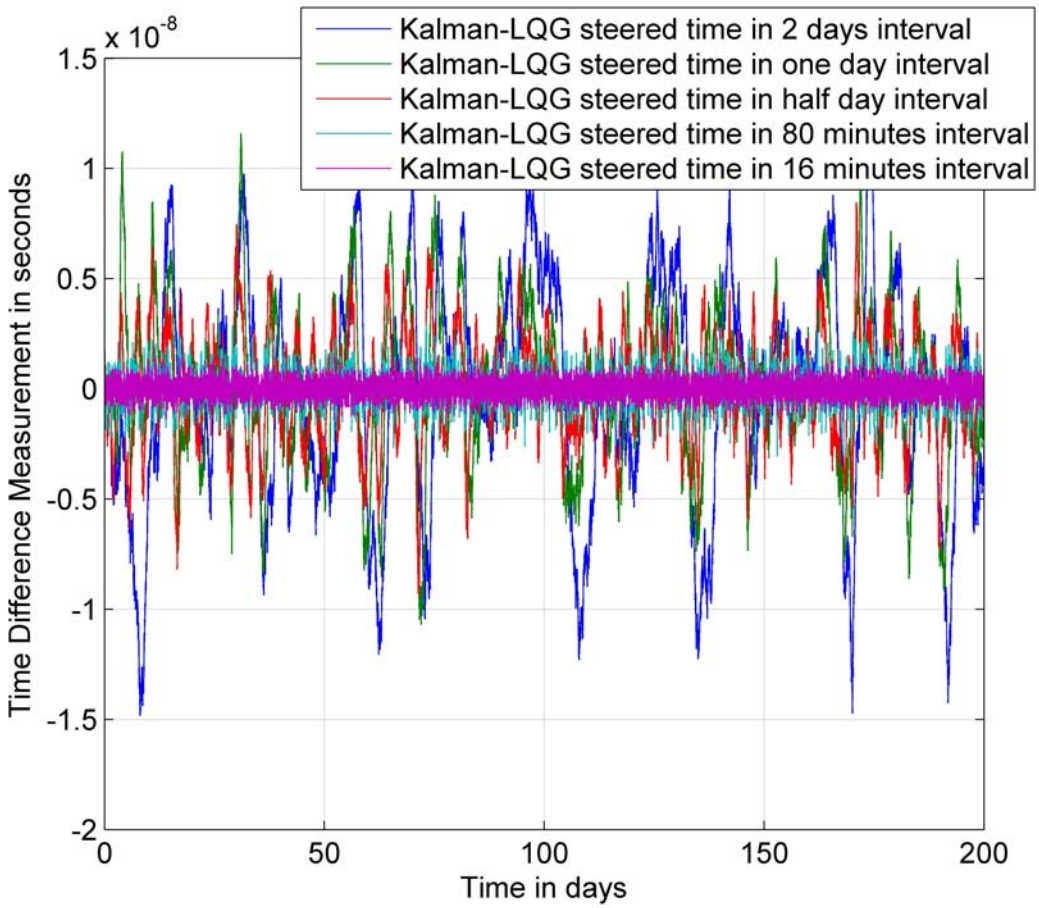

Figure 4.10: Kalman-LQG method steered time offset in different intervals

The figure shows that the Kalman-LQG method of steering is sensitive to the data rate and delay. The accuracy of the steering will increase obviously when the data arrive 'on

time'. So this method of implementation leaves much space of the future improvement.

4.4.2 INPL method steered time with different data rates

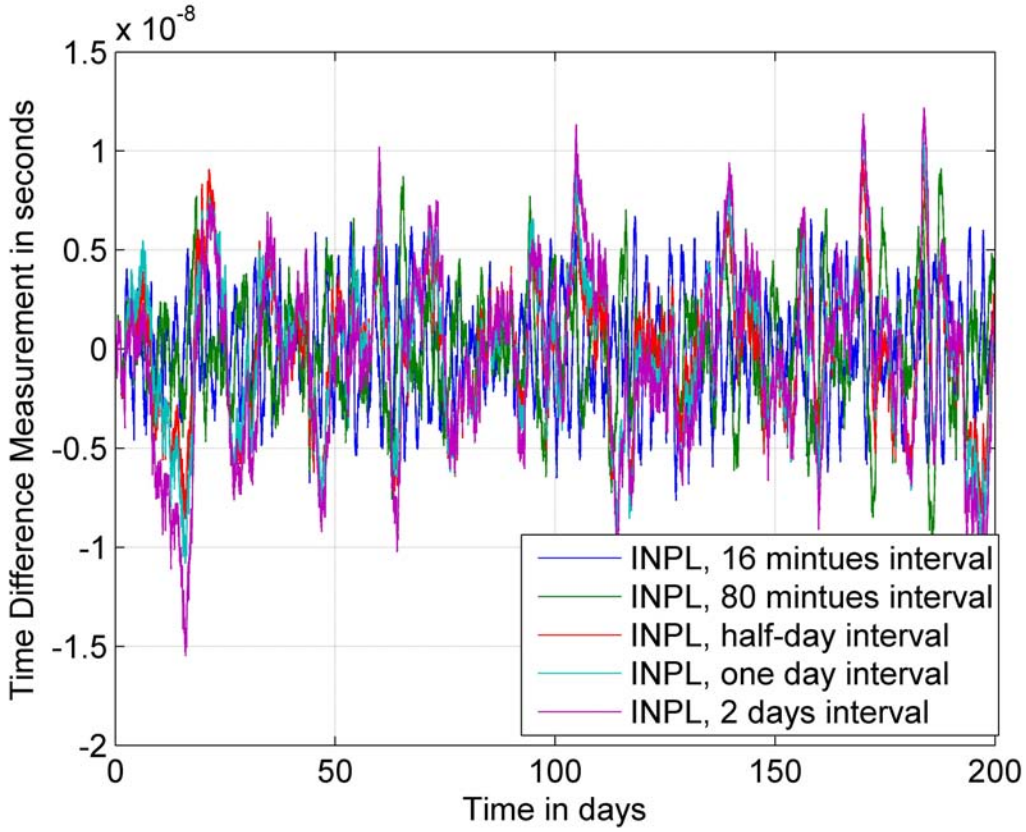

Figure 4.11: INPL method steered time offset in different intervals

The higher data rate will also bring some improvement by INPL method of steering. But this effect is not so great. By 16 and 80 minutes intervals, Kalman-LQG offers much better frequency stability.

These reasons makes the Kalman-LQG method to be chosen in the implementation.

4.4.3 Compare the Variance of the Steered Time Offset

Table 4.4 contains the variances of INPL and Kalman-LQG steered times for different data rates. According to the table, with the same data rate, the Kalman-LQG method offers smaller variance than the INPL method in steering period shorter than one day.

According to the comparisons of the two steering methods, Kalman-LQG method is not always the better one.

Interval	INPL Method	Kalman-LQG Method
Measurement	$\sigma_{Measurement}$= 3.7803×10^{-8}	
16 Minutes	σ_{INPL}= 2.6117×10^{-9}	$\sigma_{Kalman-LQG}$= 4.0668×10^{-10}
80 Minutes	σ_{INPL}= 2.8767×10^{-9}	$\sigma_{Kalman-LQG}$= 8.2672×10^{-10}
Half Day	σ_{INPL}= 3.3970×10^{-9}	$\sigma_{Kalman-LQG}$= 2.9220×10^{-9}
1 Day	σ_{INPL}= 3.7035×10^{-9}	$\sigma_{Kalman-LQG}$= 4.6588×10^{-9}
2 Days	σ_{INPL}= 4.4447×10^{-9}	$\sigma_{Kalman-LQG}$= 7.5595×10^{-9}

Table 4.4: Standard deviations of INPL and Kalman-LQG methods steered time offset in different intervals

4.4.4 Compare the Allan deviation

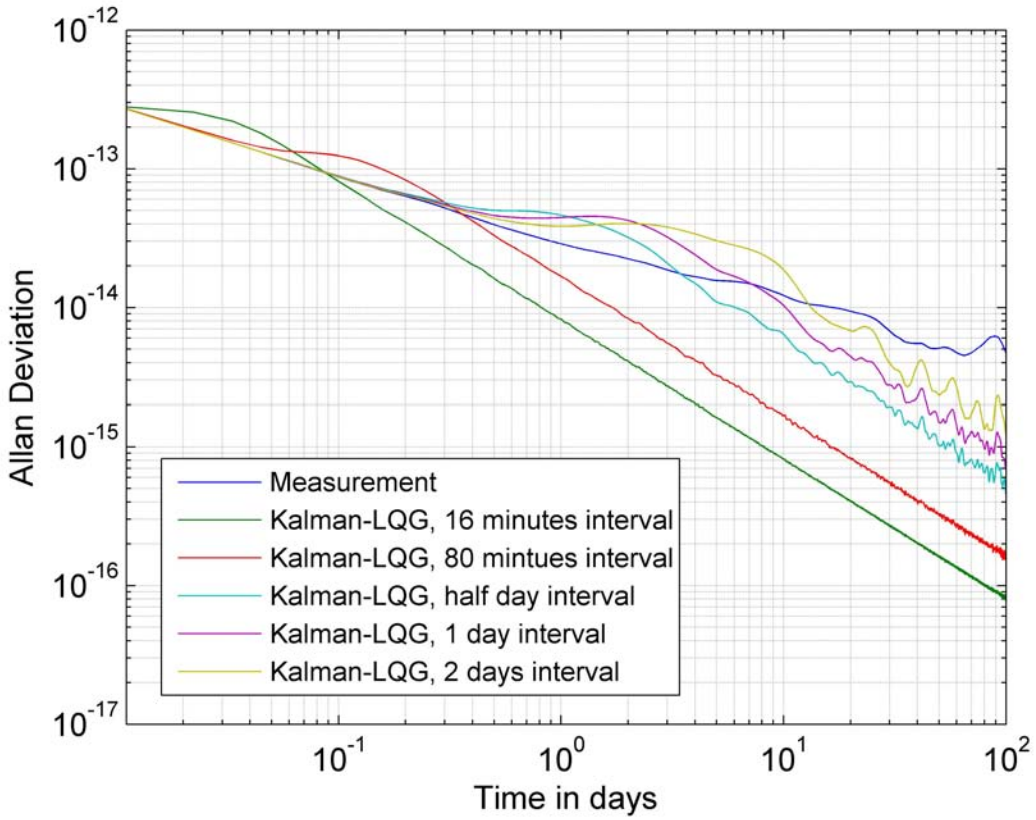

Figure 4.12: Compare the Allan deviation of Kalman-LQG steered time offset with different data rates

The Kalman-LQG method benefits from the increasing of the data rate obviously. Allan deviation decreases fast with increasing data rate. The Allan deviation of INPL steered time doesn't improve too much by increasing the data rate. Figure 4.14 shows that if the data rate is lower than half day the INPL method steered time will have better stability.

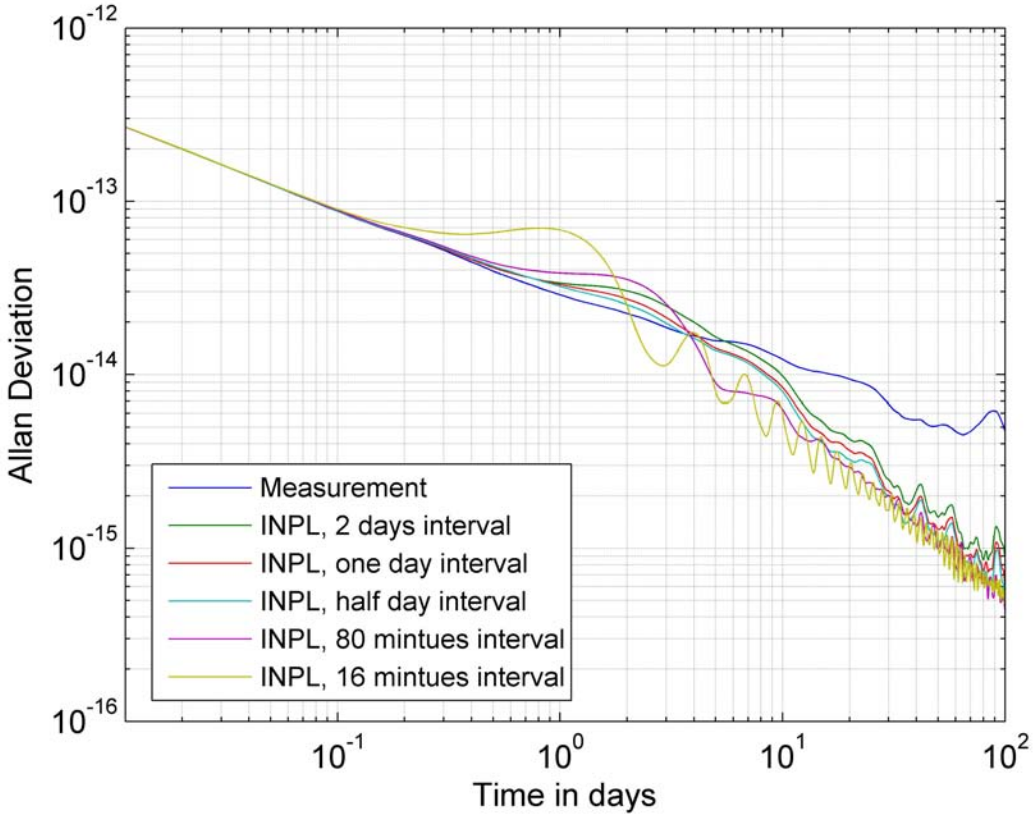

Figure 4.13: Compare the Allan deviation of INPL steered time with different data rates

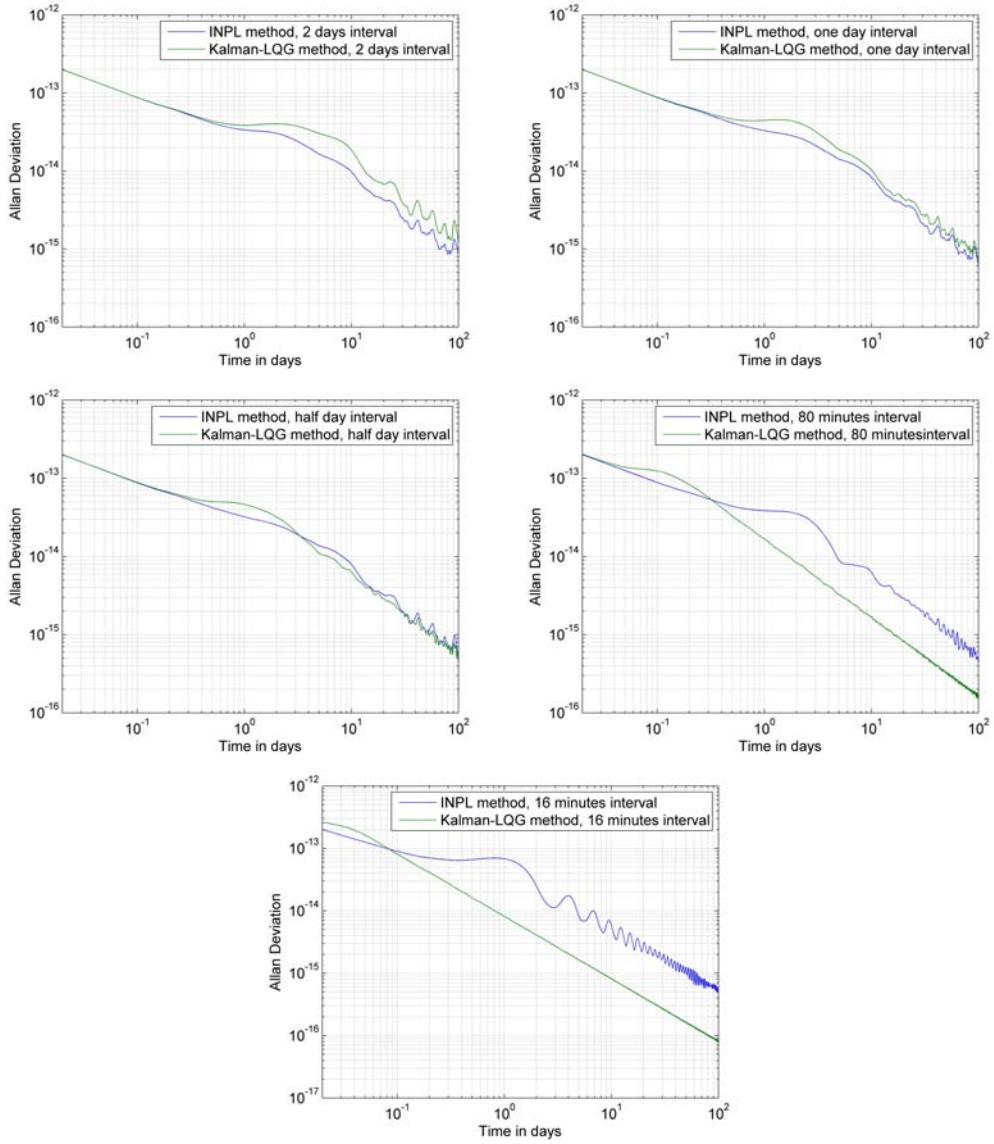

Figure 4.14: Compare the Allan deviation of INPL and Kalman-LQG method steered time with the same data rate

5 Implementation

The implementation is not only the algorithm realization, it contains also graphic user interface, communication interface, event logging, error handling and so on. In this chapter, the algorithm, equipments setup, software components, data files, graphic user interface are described.

5.1 System Description

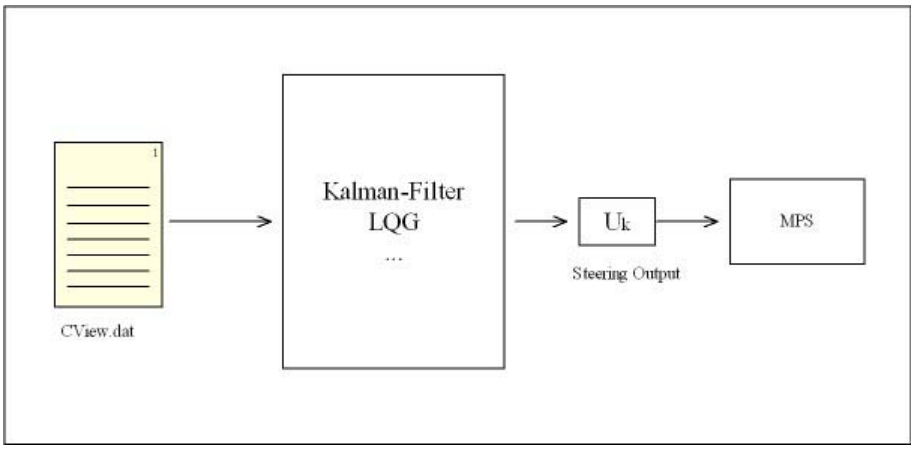

Figure 5.1: System Description

The implementation is aimed to steer the DLR time scale UTC(DLR) to UTC(PTB). One time difference data file which contains the offset of UTC(DLR) to UTC(PTB) is loaded to the system as the input. The system will then use Kalman-LQG method to calculate the steering output and send it through the RS-232 port to the micro phase shifter.

The offset data file are only available 48 hours after the offset take place. This will greatly

reduce the steering effect. In this case the Kalman-LQG method is similar as the INPL method. But the implementation Kalman-LQG method is still chosen and applied because the delay of the offset data in the future would reduce. The reduction of the delay will improve the steering effect.

Currently, the common view data between PTB and DLR is still not available. So the input data of the system is now the offset of UTC(DLR) to GPS system time which comes from the GPS time receiver.

5.2 Experimental Setup

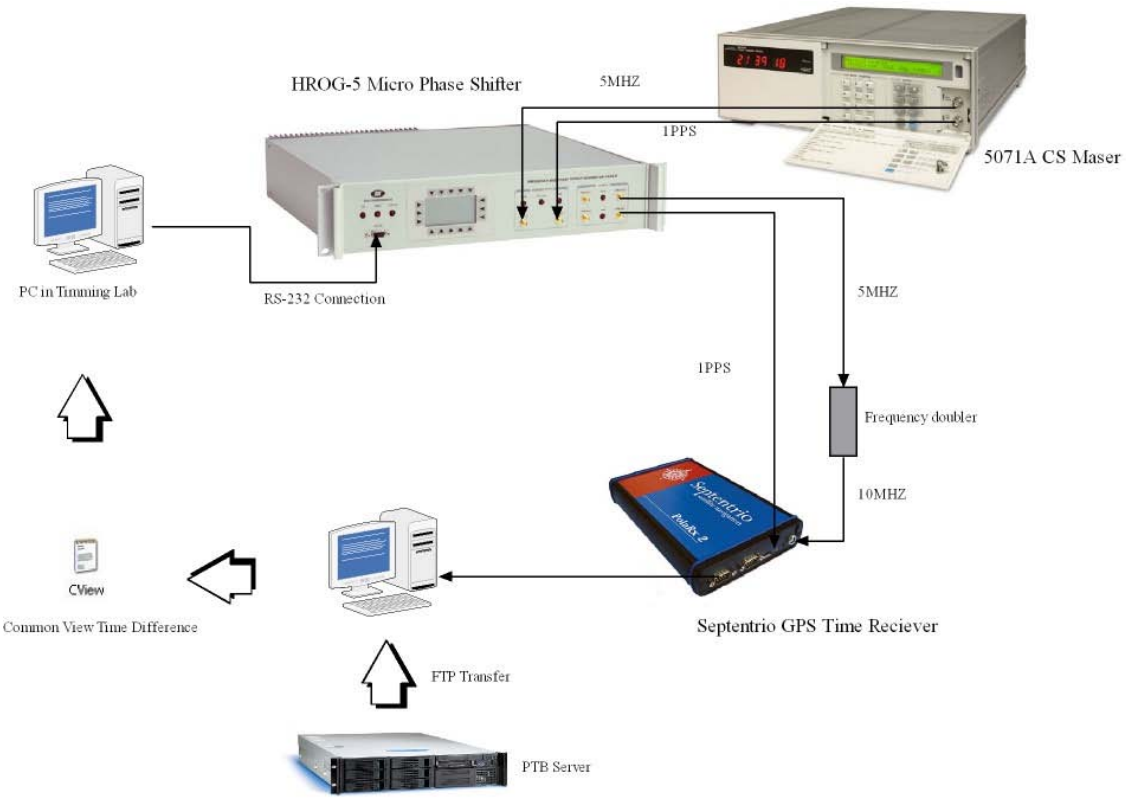

Figure 5.2: Experimental Setup

The most important equipments of the system are:

1. **5071A Primary Frequency Standard**
   Cesium Maser offers Frequency accuracy to 5 $\times 10^{-13}$

2. **HROG-5 Micro Phase Shifter**
   A high-resolution phase and frequency offset generator

3. **Septentrio GPS Time Receiver**

5071A Cesium maser is the frequency source of the system. The on board 5MHz and 1PPS output signals are connected to the HROG-5 Micro Phase shifter. HROG-5 can generate both frequency and phase offsets. The steered 5MHz and 1PPS signal will go from its output to the Septentrio GPS time receivers. Because Septentrio receivers require the 10 MHz input, one frequency doubler is connected between the HROG-5 and the GPS receiver. One PC which fetch the data from PTB via FTP transport and get the data from the Septentrio GPS receivers. The calculated time offset is stored in a file named CView.dat on the local area network. The implemented software is running on another PC which located in the timing lab. The programmes with graphic user interface calculates the steering output to the micro phase shifter from the offsets.

5.3 Design Features

According to the system setup, software algorithm, equipments communication interface and the convenience of the user the software is programmed with the following features:

1. Graphic user interface.

2. Users are allowed to change the system parameters: Q, R, QQ, RR for LQR, threshold value of the steering output.

3. Auto recover the process to the break point by restarting

4. Generate the data when Measurement is not available

5. Threshold value for steering output to filter out the very small steering signals so that to avoid the noisy steering.

6. Inform the user by E-mail and log event when very large offset detected.

7. Inform the user by E-mail and status flag when data was not available for a certain period.

8. Log the important events and data such as: initialize, reset, data fetching, processing, measurement generation, steering.

9. Show status of the micro phase shifter: time offset, frequency offset, time, data, temperature, etc.

10. Plotting data: measurement, estimation, steering.

11. Auto trigger the process every 48 hours.

12. Auto set the offset when large offset detected and the user didn't change the offset manually.

13. Setup the serial communication module to apply the steering to the micro phase shifter and read the status and the feed back from it.

5.4 Software Components

5.4.1 Software Flow Chat

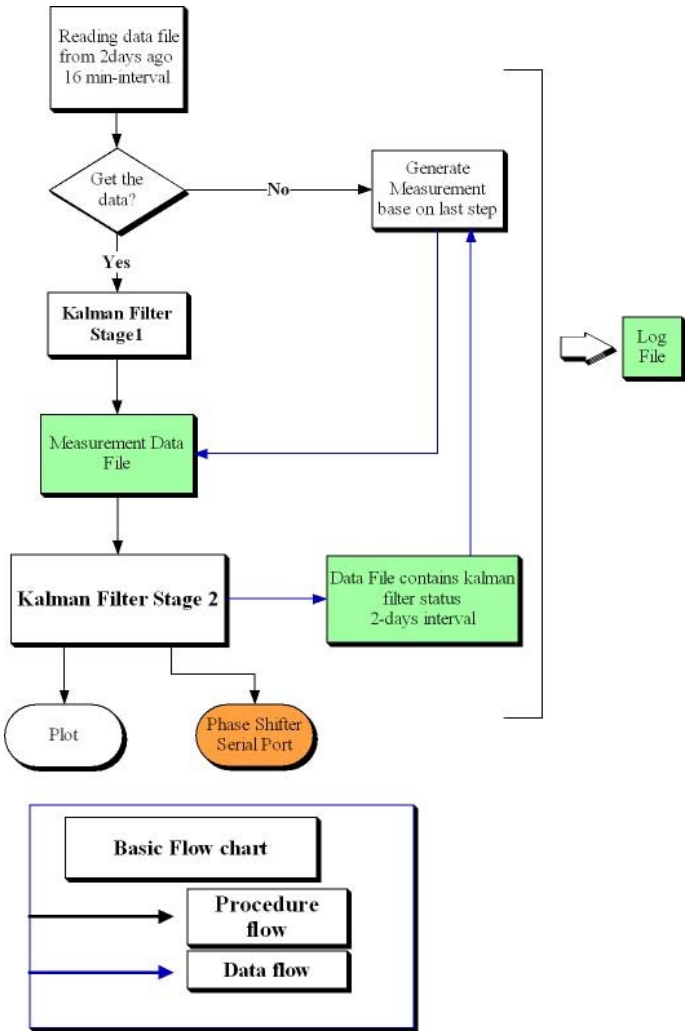

Figure 5.3: Software Flow Chat

Starting: By starting the Programme, the file last_used_parameters.txt will first be loaded. If it is not empty, then the system will load the parameters stored in it and fill the blanks on the GUI with these parameters. If it's empty, the programme will then load the default values and fill them in. The parameters contain Q,R ,QQ, RR for LQR and the threshold value for the steering output.

The user can then click the button "Start" to start the counter. The counter display shows how many seconds left before next trigger.

Execution: By the time point of trigger, the programme will try to open the common view data on the network location. The remote location is linked as a virtual local drive I on the computer.

The path of the file is like: $I : \backslash yyxxx \backslash CView.DAT$. yy is the 2 digits of year and xxx is the xxxth day of the year. For example: Oct. 5, 2006 is the 278th day of 2006. The data of that day is then stored in $I : \backslash 06278 \backslash CView.dat$.

If the file exists at the location, the data will be load in to the workspace. kalman_stage1.m starts. The 16-minutes step data are processed to get one single measure value. This value is the time difference measurement in the last 48 hours. It will be written to the file measurement.txt. If the file doesn't exist, the system will log this event as 'Data not available'. To keep the system running the programme 'generate_data.m' will then be run. It generate the possible time difference in the passed 48 hours base on the data of the last measurement and the last steering value. The estimated value will also be stored in the file 'measurement.txt'. After this, kalman_stage2.m will be called.

Kalman_stage2.m is the main estimator the software. It reads the measurement.txt and reads the parameters in the text box on the graphic user interface. After processing, it gives out the estimated value of time difference and the fractional frequency offset. By making use these two values, the Linear Quadratic Regulator will then calculate the optimal steering output u_k.

If u_k is bigger than the steering limit of micro phase shifter, the limit value will be applied to the system. If u_k smaller then the threshold value of the control output, it will be neglected. This will improve the output frequency stability.

After the estimation, in the case a very big time difference was detected(for example Z bigger than 500 nano seconds), the programm will send a warning mail automatically to

the user and log this event. The status flag on the GUI will also change to show the data availability. If the user doesn't respond to the warning mail (make manually offset)in 24 hours, the system will automatically apply the the offset to the micro phase shifter.

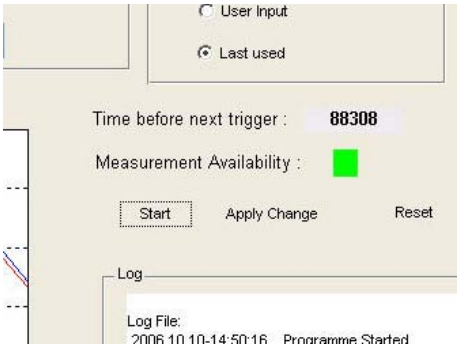

Figure 5.4: Measurement Availability indicated by lamp: green: measurement available, red: measurement not available

To ensure the system could restore the process after the restarting of the programm or reboot of the computer by power failure, all the inter parameters are stored in the file 'data.txt'. By next step of running they will be load in to the workspace again. They are also useful when try to debug the filter.

5.4.2 Programme files

The clock steering system contains the following files:

load_data.m: Fetch the time difference files from the network location.
kalman_stage1.m: Calculate the time difference of 48 hours ago from the offset before 2 days ago.
kalman_stage2.m: Predict the current time offset using the historical data and the result of kalman_stage1.m.
generate_data.m: If the data is not available on the network, this programm will predict the measurement base on the last step of measurement and last steering value.

reset_t.m: Reset the system, clear the historical data, remove the log.

initialize.m: If the system is reset by the user, the initial values will input to the Kalman filter.

serial_port.m: Acquire the status parameters of the micro phase shifter through the RS-232 port and store in a file.

plotting_t.m: Plot the measurement, time difference and frequency offset estimation and the steering output.
disp_log.m: Display the log events. **disp_serial.m**: Display the serial port status file.
disp_steer.m: Display the steer values sent to the micro phase shifter.

app_offset.m

5.4.3 GUI

GUI002.m
Graphic User Interface. The GUI will make the system user friendly. It's easy to get the results, make the modification and get the status of the system.

5.5 Data files

There are 6 data files in the system. By backup these files, the system can be recovered after the crash of the operation system or power supply failure.

Data.txt contains the inter data of the Kalman filter such as the Q, R, U, X,X_{ahead},P_{ahead}. Every time when the file kalman_stage2.m is called, the programme will first load this file and read the values of the last step.

Measurement.txt stores the measurement data. It's the output of the kalman_stage1.m and the input of the kalman_stage2.m. Each line of the file contains one single Kalman-filter filtered measurement of 48 hours ago. The last number 1 and 0 tells that if the measurement are from the data file(1) or generated by the programme(0).

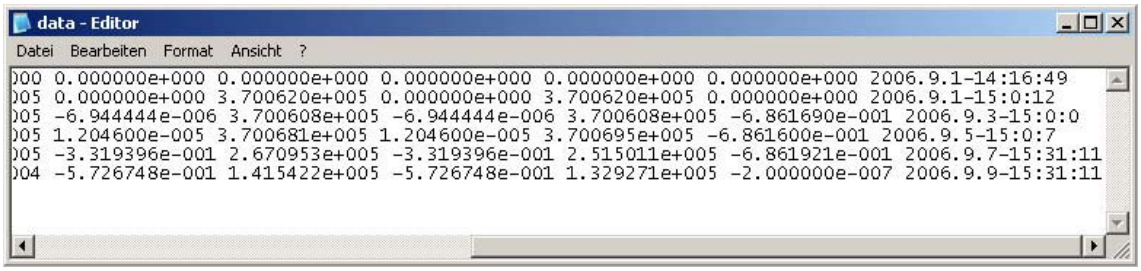

Figure 5.5: data.txt

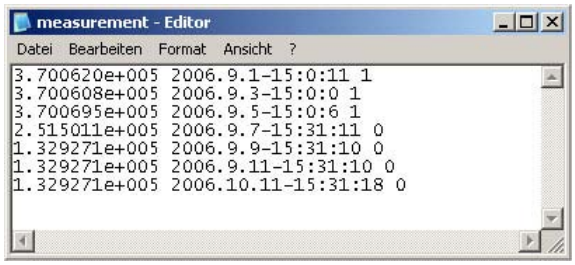

Figure 5.6: measurement.txt

serial_port1.txt stores the results of the serial_port.m. User can get the status of the micro phase shifter remotely.[1]

BAUD	the baud rate	FFOF	fractional frequency offset
TOFFS	time offset	FREQ	frequency offset
PHAS	phase offset	TEMP	instrument temperature
PPSW	1 PPS pulse width	SFFOF	last time step
SFREQ	last frequency step	SPHAS	last phase step

Serial port status updated: Because the status of the MPS is not displayed real time, so it's necessary to know the time of the status.

steering_log.txt stores all the applied steering output. The steering output to the MPS might not be the calculated value from the programme. If the calculated result is too small or too large, the alternative value will be applied. This file stores the values.

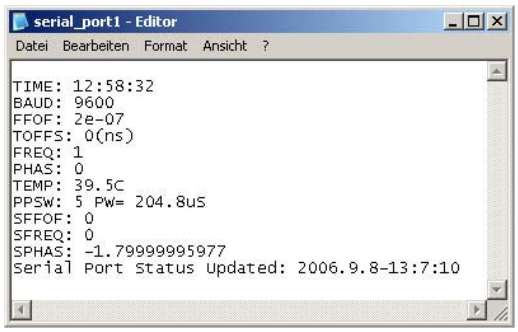

Figure 5.7: serial_port.txt

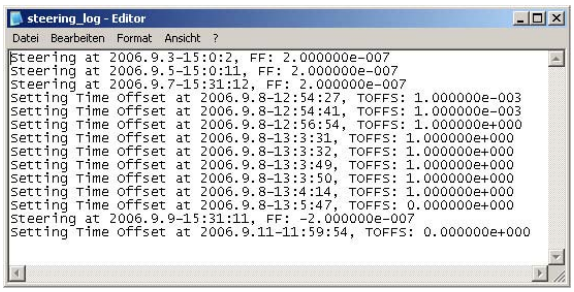

Figure 5.8: steering_log.txt

log.txt stores the system events such as resetting, initializing, data loading, processing, etc.This file could help to debug the system.

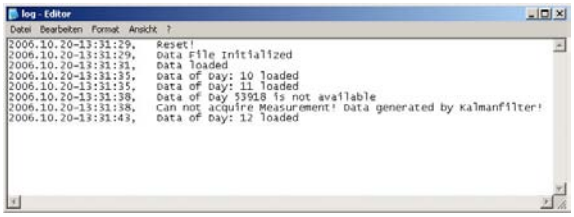

Figure 5.9: log.txt

last_used_paramters.txt stores the last used parameters or user modified parameters. By restarting, the system could restore the previous process.

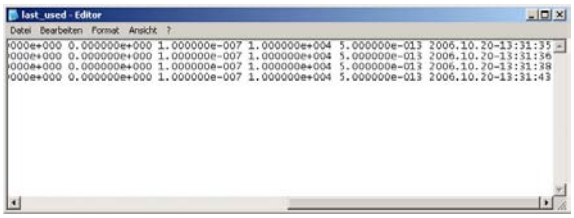

Figure 5.10: last_used_paramters.txt

5.6 Graphic User Interface

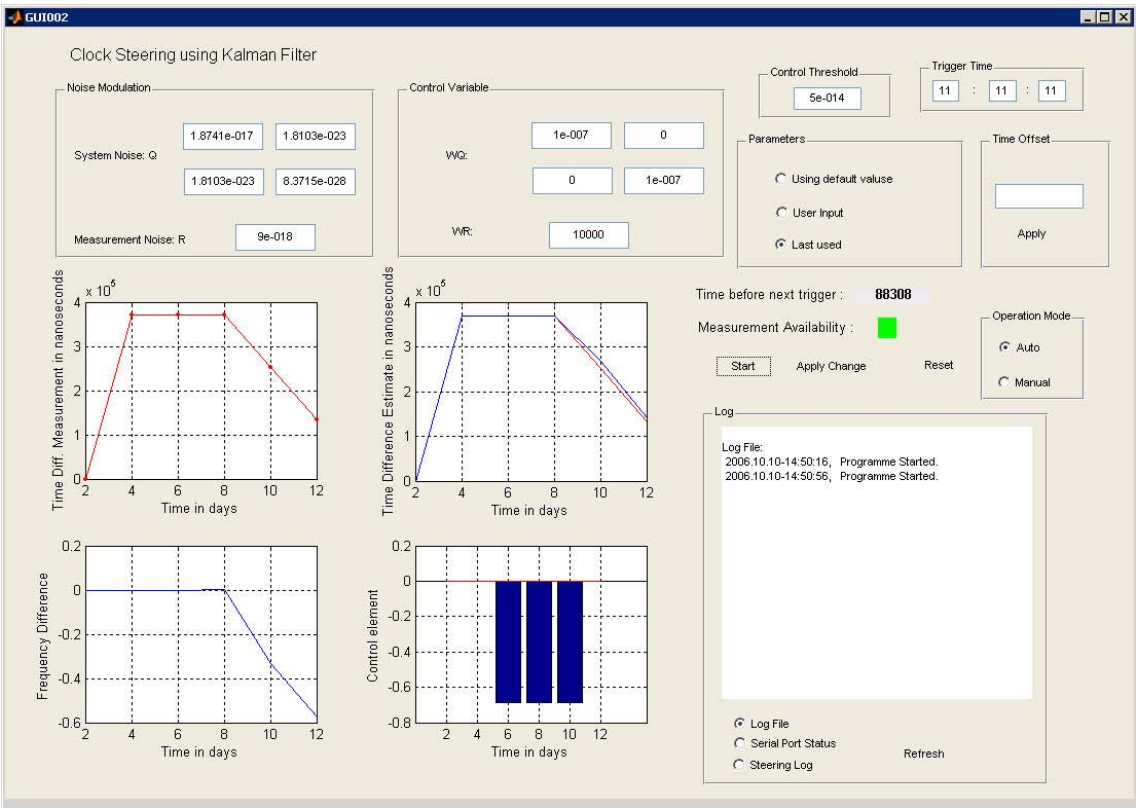

Figure 5.11: GUI

The graphic user interface makes the programme user friendly. The user can observe the data plotting directly from the GUI. User can also modify the parameters such as Q,R, WQ,WR with the mouse driver and keyboard.

The GUI are stored in the files: **GUI002.fig** and **GUI002.m**.

In the 'Trigger Time' panel, the user can insert a time to let the programme run. The default trigger time is 15:30 every 2 days.

The status flag 'Measurement Availability' tells the user if the data in the network location are available. If the data was not available for 10 days, this flag will turn red. The user

should check the GPS receiver or the network connection of the PC.

Sometimes the the system will detect a constant large offset. This might cause by the cable delay. It takes long time to cancel this offset by judge only the frequency. In this case, the user can enter a offset value in the text box 'Time Offset' and click 'Apply' with the mouse driver. The micro phase shifter will then generate a time offset with that value.

6 Summary

The aim of this diplom thesis was to design and implement the clock steering system to improve the accuracy and stability of the DLR time scale. During the work, serval filtering and steering theory were studied, such as Kalman-filter, moving average, LQG controlling.

Kalman-LQG method and INPL method were compared. The result shows that the INPL method is still a effective way of steering. It's easy to implement and in some case provide similar or even better performance than Kalman-LQG method. Kalman-LQG method is a newer method. The system noise and measurement are pre-defined. The historical data and result will help to correct the estimation and prediction. By sufficient data supply, Kalman-LQG steered time will provide better accuracy and stability.

The implementation is done with Matlab. Two stages of Kalman filter were implemented. The first stage is operated in open loop. It filters the historical data to get the final point of measurement. It's a 16 minute-interval estimator. The second stage of Kalman filter is a 2-days interval predictor. It works in close loop. It generates the control values from the one-step-ahead prediction. The control policy is a trade off between accuracy and stability by LQG. It can be adjusted by changing the parameters. All the programmes are under the graphic user interface. This makes the clock steering system easy to understand, to observe and to operate.

Because the limit of time, some ideas are not able to carry out. For example, the adaptive Kalman-filter. The artificial neural network or fuzzy theory can make the Kalman filter adaptive. The system parameters can be updated automatically during the operation. If there are more atomic masers available, it's also possible to develop a ensemble clock system. Multi-clock system enables the better reliability. The key feature of ensemble clocks is the dynamic weighting. If one or more clock doesn't work properly, its(their) weighting will be reduced. Using the above mentioned ideas, it's possible to build a relatively good time scale with the commercial frequency standards.

APPENDIX

A Symmetric Matrix

If A is a square matrix and B is symmetric matrix, then ABA^T is also a symmetric matrix.

Proving:

Let $D = A \cdot B \cdot A^T$

This means:

$$\begin{matrix} a_{11} & \cdots & \cdots & \cdots & a_{1n} \\ \vdots & a_{ii} & \cdots & a_{ij} & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & a_{ji} & \cdots & a_{jj} & \vdots \\ a_{n1} & \cdots & \cdots & \cdots & a_{nn} \end{matrix} \bullet \begin{matrix} b_{11} & \cdots & \cdots & \cdots & b_{1n} \\ \vdots & b_{ii} & \cdots & b_{ij} & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & b_{ji} & \cdots & b_{jj} & \vdots \\ b_{n1} & \cdots & \cdots & \cdots & b_{nn} \end{matrix} \bullet \begin{matrix} a'_{11} & \cdots & \cdots & \cdots & a'_{n1} \\ \vdots & a'_{ii} & \cdots & a'_{ji} & \vdots \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & a'_{ij} & \cdots & a'_{jj} & \vdots \\ a'_{1n} & \cdots & \cdots & \cdots & a'_{nn} \end{matrix}$$

Let $C = A \cdot B$

$$\begin{aligned} C_{ji} &= a_{j1} \cdot b_{1i} + \cdots + a_{ji} \cdot b_{ii} + \cdots + a_{jj} \cdot b_{ji} + \cdots + a_{jn} \cdot b_{ni} \\ &= \sum_{k=1}^{n} a_{jk} \cdot b_{ki} \end{aligned}$$

So $D = C \cdot A^T$

Similar we get:

$$\begin{aligned}
D_{ji} &= \sum_{l=1}^{n} c_{jl} \cdot a'_{li} \\
&= \sum_{l=1}^{n} [(\sum_{k=1}^{n} a_{jk} \cdot b_{kl}) \cdot a'_{li}] \\
&= \sum_{l=1}^{n} [(\sum_{k=1}^{n} a_{jk} \cdot b_{kl}) \cdot a_{il}] \\
&= \sum_{l=1}^{n} \sum_{k=1}^{n} (a_{jk} \cdot b_{kl} \cdot a_{il}) \\
&= \cdots + a_{jx} \cdot b_{xy} \cdot a_{iy} + \cdots
\end{aligned}$$

$$\begin{aligned}
D_{ij} &= \sum_{l=1}^{n} \sum_{k=1}^{n} (a_{ik} \cdot b_{kl} \cdot a_{jl}) \\
&= \cdots + a_{iy} \cdot b_{yx} \cdot a_{jx} + \cdots
\end{aligned}$$

Because B is symmetric matrix, so $b_{xy} = b_{yx}$.

So: $D_{ij} = D_{ji}$.

So: D is symmetric matrix.

B List of Abbreviation

DLR: Deutsches Zentrum für Luft- und Raumfahrt
-Germany Aerospace Center

PTB: Physikalisch-Technische Bundesanstalt
- The national standards laboratory of Germany

GPS: Global Positioning System

UTC: Coordinated Universal Time

MPS: Micro Phase Shifter

FTP: File Transfer Protocol

PPS: Pulse per Second

INPL: The national physical Laboratory of Israel

LQG: Linear Quadratic-Gaussian control

BIPM: Bureau international des poids et mesures
-International Bureau of Weights and Measures(France)

TWSTFT: Two-way Satellite Time and Frequency Transfer

CGGTTS: CCTF Group on GNSS Time Transfer Standards

CCTF: Consultative Committee for Time and Frequency

TFL: Time and Frequency Limited

NBS: National Bureau of Standards(USA)

MSE: Mean Square Error

C List of mathematical symbols:

C.1 Values related to Kalman Filter

$\hat{X}(k)$: state estimation
$\hat{X}_{k/k-1}$: one-step-ahead prediction
P_k: estimate error covariance
$P_{k/k-1}$: one-step-ahead prediction error covarince
u_k: steering output
K: Kalman Gain
Q: system noise covariance
R: measurement noise covariance
Z: measurement
H: observation model
W_k: process noise
V_k: observation noise
ϕ: state transition matrix
Γ: system noise drive matrix

C.2 Values related to LQG

$V_t(z)$: minimum cost starting from state t
$V_N(z)$: final state cost at state N
J_U: quadratic cost function
w^*: optimal control
Q: relative weight of state deviation
R: relative weight of input usage

C.3 Values related to INPL method

SSD: squared second(the unit of time) difference
$\hat{Y}_{io}(t)$: predicted frequency offset
$\hat{X}_{io}(t)$: predicted phase offset
E_t: filtered squared second(the unit of time) difference
W: weight for each clock
N_r: weight of exponential filter
X_{os}: steering phase add to the software phase
X_{utc}: system time respect to UTC
X_{sutc}: the steered output with respect to UTC
Y_s: changing rate of the X_{os}

C.4 Other Values

$S_y(f)$: noise spectral density
$\sigma^2_{y0}(\tau)$: Allan Variance
h_2: coefficient for white phase noise
h_1: coefficient for flicker phase noise
h_0: coefficient for white frequency noise
h_{-1}: coefficient for flicker frequency noise
h_{-2}: coefficient for random walk frequency noise
$R_{xy}(t, \tau)$: cross-correlation function between two processes
$\hat{G}_0(f)$: gain to compute the optimal control

Bibliography

[1] *High Resolution Offset Generator HROG-5 Operating Manual.*

[2] J.B.McGraw A.J.Van Dierendonck and R.Grover Brown. Relationship between allan variances and kalman filter parameters. In *Proc. 16th Precise Time and Time Interval Conference*, 1984.

[3] A.Shenhar, A.Lepek W.Litman, D.W.Allan A.Citrinovitch, and Peppler T, K. Israel's new synchronized time scal, utc(inpl). In *42nd Annual Frequency Control Symposium*, volume 78, page 78, 1988.

[4] J.A. Barens and Allan D.W. A statistical model of flicker noise. In *Proceedings of the IEEE, Vol. 54*, 1966.

[5] Stephen Boyd. Linear quadatic regulator: Discrete-time finite horizon. In *Linear Dynamical Systems*. Stanford University Online Course Material, 2005.

[6] David B.Wagner. Power programming-dynamic programming. *The Mathematica Journal*, Vol.5:Issue 4, 1995.

[7] Yongyi Chen Congwei Hu, Wu Chen and Dajie Liu. Adaptive kalman filtering for vehicle navigation. *Journal of Global Positioning System*, 2:42–47, 2003.

[8] D.A.Howe nad F.L.Walls D.B.Sullivan, D.W.Allan. *Characterization of Clocks and Oscillators*. National Institute of Standards and Technology, 1990.

[9] D.W.Allan F.B.Varnum, D.W.Brown and T.K.Peppler. Comparison of time scales generated with the nbs ensemble algorithm. *19th Precise Time and Time Interval symposium*, 12:12, 1987.

[10] Carlos Felippa. Matrix calculus. In *Introduction to Finite Element Methods*. University of Colorado at Boulder online Course Material, 2006.

[11] Claude Audoin & Bernard Guinot. *The Measurement of Time*. Cambridge University Press, 2001.

[12] Jonathan How. Feedback control. In *Feedback Control System*. MIT Open Course Ware, 2001.

[13] Jianye Liu Jing Bai and Xin Yuan. Study of fuzzy adaptive kalman filtering technique. *Information and Control*, 31:193–197, 2002.

[14] J.Rutman and F.L.Walls. Characteristics of frequency stability in precision frequency sources. In *Proc. IEEE*, 1991.

[15] Paul Koppang and Robert Leland. Linear quadratic stochastic control of atmoic hydrogen masers. In *Proc. IEEE*, 1999.

[16] M.Oussalah and J.De Schutter. Adaptive kalman filter for noise identification. In *25th International Conference on Noise & Vibration Engineering*, 2000.

[17] Kartaschoff P. *Frequency and Time*. London:Academic Press, 1978.

[18] P.Kopand and R.Leland. Steering of frequency standards by the use of linear quadratic gaussian control theory. In *27th Precise Time and Time Interval symposium*, 1995.

[19] R.G.Brown and P.Y.C.Hwang. *Introduction to Random Signals and Applied Kalman Filtering, 3rd ed.* New York:Wiley, 1992.

[20] wikipedia, $http://en.wikipedia.org/wiki/Allan\_variance$. *Allan Variance*.

[21] wikipedia, $http://en.wikipedia.org/wiki/Dynamic\_programming$. *Dynamic Programming*.

[22] wikipedia, $http://en.wikipedia.org/wiki/Kalman\_filter$. *Kalman Filter*.

[23] wikipedia, $http://en.wikipedia.org/wiki/Moving\_average$. *Moving Average*.

[24] wikipedia, $http://en.wikipedia.org/wiki/Symmetric\_matrix$. *Symmetric Matrix*.

[25] H.F.Fligel W.J.Klepczynski and D.W.Allan. Gps time steering. In *19th Precise Time and Time Interval symposium*, 1986.

[26] W.J.Riley. Allan variance. In *The Basics of Frequency stability anallysis*. Hamilton Technical Services, 2000.

[27] Hongyue Zhang Yongyuan Qin and Shuhua Wang. *Kalman Filtering and Combined Navigation*. Xian:Northwestern Ploytechnical University, 1998.