Real-time FPGA-based Kalman Filter for Constant and Non-constant Velocity Periodic Error Correction

Chen Wang[a,*], Ethan D. Burnham-Fay[b], Jonathan D. Ellis[b,c]

[a] *Department of Electrical and Computer Engineering*
[b] *Department of Mechanical Engineering*
[c] *The Institute of Optics*
University of Rochester, Rochester, NY 14627, USA

Abstract

Displacement measuring interferometry has high resolution and high dynamic range, which is widely used in displacement metrology and sensor calibration. Due to beam leakage in the interferometer, imperfect polarization components, and ghost reflections, the displacement measurement suffers from periodic error, whose pitch is multiple harmonics of the Doppler frequency. In dynamic measurements, periodic error is usually on the order of nanometers, which impacts the dynamic measurement accuracy. This paper presents an approach to estimate and correct periodic error in real time based on an extended Kalman filter, which has the capability to deal with both constant and non-constant velocity motions. This algorithm is implemented on an application-specific hardware architecture in an FPGA, which has advantages in throughput and resource usage compared with conventional implementations. The measurement validation shows that this approach can effectively eliminate the periodic error for both constant and non-constant velocity motion, and the residual error reaches to the level of the background noise of the interferometer.

Keywords: Displacement Measurement, Field Programmable Gate Arrays, Interferometry, Kalman Filter, Periodic Error
2010 MSC: 00-01, 99-00

1. Introduction

Displacement measuring interferometry (DMI) is a widely used technique for displacement metrology, and for position feedback sensing in, *e.g.*, photolithographic steppers, and position sensor calibration for, *e.g.*, capacitance sensors, linear variable differential transducers (LVDTs), and linescales.

*Corresponding author. Tel.: +1 585 275 9343; fax: +1 585 256 2509.
Email addresses: chen.wang@rochester.edu (Chen Wang), ethan.burnham-fay@rochester.edu (Ethan D. Burnham-Fay), j.d.ellis@rochester.edu (Jonathan D. Ellis)

Interferometer architectures can vary depending on the desired target geometry and intended application [1]. There are two types of displacement interferometers: 1) homodyne, meaning a single optical frequency is used, and 2) heterodyne, meaning two optical frequencies are used. In all cases, the metrology principle is the same: target position changes are recorded as a measured phase change. So the target displacement is determined by the measured phase change, the nominal relationship between them is

$$x = \frac{\lambda}{Nn}\frac{\phi}{2\pi}, \tag{1}$$

where x is the displacement of the target, N is the interferometer fold factor (two for the interferometer used in this work), n is the refractive index along the optical path difference, λ is the wavelength of the light in the measurement arm, and ϕ is the phase difference between the reference and measurement signals.

Displacement interferometry is often used in applications that require its high resolution and high precision in both static and dynamic measurement instances. There are many optical, mechanical, and electrical error sources in the interferometer system, which must be avoided or corrected to achieve better performance [2]. Existing machine tool standards [3, 4, 5] define quasi-static calibration procedures, which are often performed with laser interferometers. However, there is a push to establish a standard for performing dynamic stage calibrations to complement the existing standards for quasi-static calibrations [6]. For dynamic calibrations, periodic error can have different implications for the measurement error in static versus dynamic cases. In this paper, we propose a Kalman filter based method to estimate and correct periodic error in real time. We detail the algorithm, demonstrate its functionality in simulated results, and validate its real-time performance using a precision positioning system moving with both constant and non-constant velocity motion profiles.

2. Periodic Error

Periodic error is a non-cumulative error that manifests as a deviation from the linear assumption between displacement and phase in (1). Optical and electrical errors can lead to periodic error, causing a linear displacement to exhibit a non-linear phase change. This non-linear phase change comprises the nominal linear relationship between phase and displacement plus a number of sinusoidal harmonics, whose pitch is a multiple of the optical resolution of the interferometer. Many previous studies on periodic error source investigation have been done, *e.g.*, [7, 8]. For all interferometers, beam leakage in the interferometer, imperfect polarization components, and ghost reflections can all cause periodic errors. For heterodyne interferometers, the laser source can suffer from polarization mixing, leading to frequency mixing in the source [9, 7, 8]. The aforementioned leakage, imperfections, and ghost reflections then exacerbate the effects of laser source mixing. For homodyne interferometers, the

aforementioned leakage, imperfect optics, and ghost reflections get exacerbated on the detection side of the interferometer. Here, the overlapping, but not interfering, arms of the interferometer are split to generate an additional phase shift in one set of beams to generate in-phase and quadrature detection signals. In either case, homodyne or heterodyne, the effect of periodic error is similar in form and magnitude.

There are two common ways to correct or compensate the periodic error to achieve more precise measurement: 1) design a novel optical configuration that eliminates the periodic error inherently [10, 11, 12, 13, 14], and 2) use a compensation algorithm applied to the measurement data to correct the periodic error electronically [15, 16, 17, 18, 19, 20, 21, 22, 23, 24]. A novel optical configuration of interferometer would lead vastly large and complicated setup, and be more sensitive to environment. While a signal processing method could potentially apply to most of the traditional-configured interferometers, one can just upgrade the electronic systems without changing any optical configuration. In this work, we investigate and improve the implementation of a signal processing algorithm to correct periodic error, achieving on-line estimation and real-time correction.

Previous work has demonstrated an effective model for the contributing sources of periodic error [25, 26, 27, 28]. The measurement and reference signals can be modeled as

$$\begin{aligned} I_m \propto & \cos(2\pi f_s t + \phi) + \gamma_{11}\cos(2\pi f_s t) + \gamma_{12}\sin(2\pi f_s t) \\ & + \gamma_{21}\cos(2\pi f_s t - \phi) + \gamma_{22}\sin(2\pi f_s t - \phi), \end{aligned} \tag{2}$$

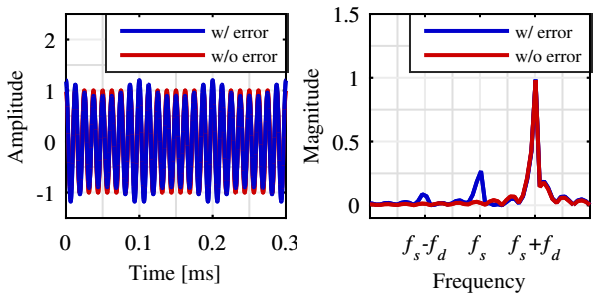
$$I_r \propto \cos(2\pi f_s t), \tag{3}$$

where f_s is split frequency, ϕ is the displacement-induced phase difference between measurement and reference signals, which is the desired measurement quantity. In (2), the first term is nominal measurement signal, the second and third terms are the first-order periodic error, whose period is one cycle per fringe, and the fourth and fifth terms are second-order periodic error, whose period is two cycles per fringe. Amplitudes γ_{11}, γ_{12}, γ_{21}, and γ_{22} are corresponding coefficients of the periodic errors. Typically, γ_{11} and γ_{21} contribute major periodic error, and value of γ_{12} and γ_{22} are very small, which are introduced by slight phase drift. Figure 1 illustrates how the extra periodic error terms in measurement signal influence its amplitude and spectrum.

To extract the phase difference ϕ, lock-in detection and single-bin discrete Fourier transform methods are commonly used [1, 29]. In those methods, the measurement signal is multiplied with the in-phase and quadrature components of the reference signal. Then, the in-phase and quadrature components pass low-pass filters to remove the high-frequency components, resulting in

$$I_x \propto +\cos(\phi) + \gamma_{11} + \gamma_{21}\cos(-\phi) + \gamma_{22}\sin(-\phi), \tag{4}$$

$$I_y \propto -\sin(\phi) + \gamma_{12} - \gamma_{21}\sin(-\phi) + \gamma_{22}\cos(-\phi). \tag{5}$$



(a) (b)

Figure 1: The measurement signal I_m in time domain (a) and frequency domain (b) when simulating a constant velocity motion. In the time domain, the amplitude of the measurement signal is varied, due to the appearance of periodic error. In the frequency domain, two extra peaks appear, represent first- and second-order periodic error respectively, whose frequencies are related to the Doppler frequency f_d, and split frequency f_s [27].

Like terms can be combined, yielding

$$I_x = \sqrt{(1+\gamma_{21})^2 + {\gamma_{22}}^2}\cos(\phi+\varphi_1) + \gamma_{11}, \tag{6}$$

$$I_y = -\sqrt{(1-\gamma_{21})^2 + {\gamma_{22}}^2}\sin(\phi-\varphi_2) + \gamma_{12}, \tag{7}$$

where $\tan\varphi_1 = \gamma_{22}/(1+\gamma_{21})$ and $\tan\varphi_2 = \gamma_{22}/(1-\gamma_{21})$. These terms can be expressed in the generic form [19],

$$I_x = a\cos(\phi_g+\varphi) + I_{xc}, \tag{8}$$

$$I_y = b\sin(\phi_g) + I_{yc}, \tag{9}$$

where

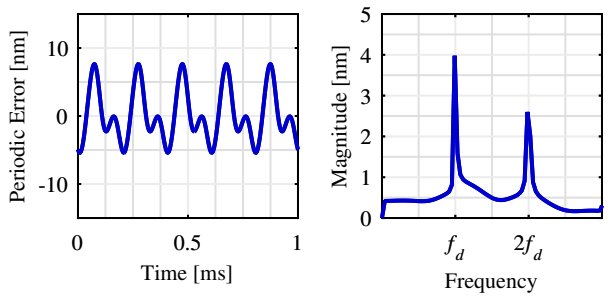
$$\phi_g = \phi - \varphi_2,$$

$$\varphi = \varphi_1 + \varphi_2,$$

I_{xc} and I_{yc} are the DC offsets, a and b are the gain of the AC components, and φ is the quadrature phase shift of ϕ_g (not exactly $\pi/2$ shift between I_x and I_y). The displacement is represented by the change of phase ϕ, with constant offset φ_2 will not impact the change of ϕ, thus, not affect displacement measurement. The in-phase I_x and quadrature I_y signals for homodyne interferometers have the same expression [17], thus the following correction method works for both heterodyne and homodyne interferometers.

In an ideal interferometer setup, $\gamma_{11}, \gamma_{12}, \gamma_{21}, \gamma_{22} = 0$, so $|a| = |b|$, $I_{xc} = I_{yc} = 0$ and $\varphi = 0$, thus the phase difference ϕ can be extracted by an arctangent operation directly,

$$\phi_g = \arctan\left(\frac{I_y}{I_x}\right) = \arctan\left(\frac{\sin(\phi_g)}{\cos(\phi_g)}\right). \tag{10}$$



(a) (b)

Figure 2: The periodic error of displacement in time domain (a) and frequency domain (b) when simulating a constant velocity motion. In time domain, the overall periodic error is usually on the order of nanometer. In frequency domain, the two orders of periodic error locate at Doppler frequency and two times of Doppler frequency, respectively. If the motion is non-constant velocity, the two peaks will keep shifting on the spectrum. It is impractical to use Fourier transform to characterize the periodic error in a long period.

However, in practice, the DC offsets and quadrature phase shift are non-zero, and the gains are unequal. If (10) is used to extract the phase, the phase it will be affected by periodic error, which is illustrated in Figure 2. To accurately determine the phase including periodic error,

$$\phi_g = \arctan\left(\frac{I_y - I_{yc}}{(I_x - I_{xc})\cdot\alpha + (I_y - I_{yc})\cdot\beta}\right), \tag{11}$$

where α and β are correction coefficients in terms of a, b, and φ.

$$\alpha = \frac{b}{a\cos(\varphi)}, \tag{12}$$

$$\beta = \tan(\varphi). \tag{13}$$

3. Correction Algorithm

Several data processing algorithms have been proposed to correct the periodic error. Among them, ellipse fitting is an essential correction method in interferometer applications. Heydemann [15] presents a linear least-squares method to identify and correct the gain ratio b/a, DC offsets, and quadrature phase shift in (8,9) by fitting the ellipse, then calculate the phase using the corrected reference and measurement signals. Birch [16] improves the method and uses a computer to simulate this correction method.

Some modern data processing techniques have also been applied to periodic error correction, such as neural networks [30] and wavelet transforms [31]. However, due to their computational complexity, they are more suitable for off-line processing, and not straightforward to implement an on-line, real-time solution.

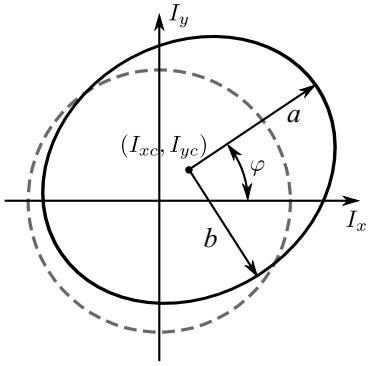


Figure 3: The Lissajous curve of raw (I_x, I_y) (ellipse) and corrected (I_x, I_y) (circle). For those coefficients in (8,9), I_{xc} and I_{yc} are the center of ellipse, a and b are major and minor semi-axes, and φ is the angle between the X-axis and the major axis. The goal of ellipse fitting is to eliminate the influence of DC offset, unequal gain and quadrature phase shift.

3.1. Ellipse Estimation

Mathematically, the Lissajous curve of (I_x, I_y) is an ellipse (see Figure 3). In analytic geometry, the point (I_x, I_y) satisfies

$$AI_x{}^2 + BI_xI_y + CI_y{}^2 + DI_x + EI_y + F = 0, \tag{14}$$

where the coefficients (A, B, C, D, E, F) determines the ellipse. Since the trace $A+C$ can never be zero for an ellipse, the arbitrary scale factor can be removed by the normalization $A + C = 1$. Thus, the ellipse can be described by a vector (A, B, D, E, F) [32]. By estimating the ellipse coefficients, the correction coefficients α, β, I_{xc} and I_{yc} can be calculated using (A, B, D, E, F). Then, the periodic error can be corrected by the correction coefficients.

Estimating the ellipse coefficients from noisy data is a common problem in the field of computer vision. Several techniques for ellipse fitting have been presented, including a linear least-squares, orthogonal least-squares, gradient-weighted least-squares, bias-corrected renormalization, Kalman filters, and robust techniques [33]. Some of them have been adjusted and applied to periodic error correction, including linear least-squares and Kalman filtering.

Wu, *et al.* [17] applied a linear least-squares method on a set of (I_x, I_y) measurement data. A computer was used to post-process the estimation and perform correction operations. Because of the off-line processing, the corrected data is not suitable for further real-time measurement and control use.

Eom, *et al.* [20] presented an implementation to adjust (I_x, I_y) in real time by using a tunable analog circuit. However, the correction coefficients still must be calculated on a computer using preliminary measurement data. Each time the optical setup changes, the correction coefficients must be recalculated off-line. Also, the correction coefficients must be treated as constant during one measurement period. Furthermore, the analog circuit can potentially introduce other noise and drift.

Chawah, *et al.* [23] established a system to estimate the coefficients and correct the error based on a Kalman filter on a computer. However, the measurement data has to be transferred from the data acquisition system (FPGA) to the computer for off-line processing. Like Wu's method, this too is unsuitable for real-time measurement and control.

Kim, *et al.* [21] developed a digital signal processing module for real-time periodic error compensation. It can continuously obtain a specific phase result at the multiples of $\pi/4$ to estimate the correction coefficients through simple arithmetic calculations, instead of matrix operations. However, it is easily affected by the accuracy of those specific phases and the speed of the phase change.

Wang, *et al.* [18] designed a real-time error correction technique based on a microcontroller and computational analog circuit. It uses microcontroller run a regression analysis to calculate the correction coefficients. It can perform on-line updates to the correction coefficients, but at a low frequency, only 15 Hz. Also, it uses a computational analog circuit to correct the error, but has limited bandwidth about 100 kHz.

From the above investigations, a linear least-square method usually leads to a closed-form solution, however, it must wait for a large set of measurement data to be available, and then operate a single joint evaluation, which is a set of matrix operations and is difficult to run in real-time. Unlike a linear least-square method, a Kalman filter has a recursive nature, which makes it is suitable for processing data in a time series [33]. However, it also has several matrix operations where in previous work it was chosen to do off-line processing using a computer, which also makes the real-time, on-line ellipse estimation impossible.

In this paper, we present an approach to on-line estimate and update correction coefficients and real-time correct the periodic error based on a Kalman filter, which is capable of self-adjusting to both homodyne and heterodyne interferometers and even correct time-varying periodic error. In this work, we developed a compact phasemeter system integrating data acquisition, phase demodulation, coefficient estimation, and periodic error correction all in one FPGA board.

3.2. Kalman Filter

The Kalman filter is a minimum mean-square error estimator, which gives the optimal estimation of a linear dynamic system model [34]. A discrete linear dynamic system can be expressed by

$$\mathbf{x}_k = \mathbf{T}_k \mathbf{x}_{k-1} + \mathbf{c}_k + \mathbf{w}_k, \tag{15}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_{k-1} + \mathbf{n}_k. \tag{16}$$

where the $\mathbf{x}_k$ is the current state vector, $\mathbf{T}_k$ is the state transition model which is applied to the previous state $\mathbf{x}_{k-1}$, $\mathbf{c}_k$ is the control vector; $\mathbf{w}_k$ is the process noise and usually modeled as white noise, $\mathbf{z}_k$ is the current observation (or

measurement) of current state $\mathbf{x}_k$, $\mathbf{H}_k$ is the observation model, and $\mathbf{n}_k$ is the observation noise.

In this design, the Kalman filter is applied to estimate the ellipse coefficients (A, B, D, E, F), so the state vector $\mathbf{x}$ is $(A, B, D, E, F)^T$. The ellipse coefficients should be the same for all points on the ellipse, so (15) can be simplified by setting $\mathbf{T}_k$ as identity vector and set $\mathbf{w}_k$ and $\mathbf{c}_k$ as zero, thus,

$$\mathbf{x}_k = \mathbf{x}_{k-1}.$$

However, the observation model is nonlinear for ellipse estimation, so (16) should be rewritten in form

$$\mathbf{z}_k = h\left(\mathbf{x}_{k-1}\right) + \mathbf{n}_k,$$

where $h\left(\right)$ is the observation model of ellipse estimation,

$$h\left(\mathbf{x}, I_x, I_y\right) = A{I_x}^2 + BI_xI_y + (1-A){I_y}^2 + DI_x + EI_y + F.$$

To still use the Kalman filter methodology [34] to estimate ellipse coefficients, an extended Kalman filter (EKF) should be used to handle this nonlinear problem. The basic thought is that it linearizes the observation model at each point using a Taylor series expansion (akin to a tangent line at a certain point on a curve), then uses the linearized model to replace the original model in computation.

The following details the principle of ellipse estimation using an extended Kalman filter [32]. It linearizes models at each new value of (I_x, I_y). Because the tangent lines at different points on the ellipse are different, the linearized observation models and the covariance of observation noise $\mathbf{n}_k$ must be updated each time using

$$\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}_k} = \left({I_x}^2 - {I_y}^2, I_xI_y, I_x, I_y, 1\right), \tag{17}$$

$$\begin{aligned}\mathbf{R}_k &= \Sigma^2\left(\left(\frac{\partial h}{\partial I_x}\right)^2 + \left(\frac{\partial h}{\partial I_y}\right)^2\right)\\ &= \Sigma^2\left(\left(2AI_x + BI_y + D\right)^2 + \left(BI_x + (1-A)I_y + E\right)^2\right),\end{aligned} \tag{18}$$

where $\mathbf{H}_k$ is linearized observation model, $\mathbf{R}_k$ is covariance of observation noise $\mathbf{n}_k$, and Σ is the noise level.

Then the model enters the Predict Step of the Kalman filter, it uses the previous state to estimate the current state. Two variables should be estimated, one is state $\mathbf{x}_{k|k-1}$, and another is error covariance $\mathbf{P}_{k|k-1}$, which measures the estimated accuracy of the state estimate. As it stated before, the $\mathbf{T}_k$ is identity vector, the formulas in the Predict Step are

$$\mathbf{x}_{k|k-1} = \mathbf{x}_{k-1|k-1}, \tag{19}$$

$$\mathbf{P}_{k|k-1} = \mathbf{P}_{k-1|k-1}. \tag{20}$$

The initial value of $\mathbf{P}_k$ is an identity matrix.

After that, it enters the Update Step, which uses current observation information to adjust the estimated state. The current observation information is the measurement residual $\mathbf{y}_k$, which is the difference between the true observation and estimate measurement. The purpose of the Kalman filter is to use measurement residual to refine the estimate state, and then make the measurement residual in following iterations approach zero, thus, a feedback loop is built. The formulas in the Updated Step are

$$\mathbf{y}_k = \mathbf{z}_k - h(\mathbf{x}_{k|k-1}, I_x, I_y), \tag{21}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T / (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k), \tag{22}$$

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k \mathbf{y}_k, \tag{23}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\, \mathbf{P}_{k|k-1}, \tag{24}$$

where $\mathbf{K}_k$ is optimal Kalman gain, $\mathbf{x}_{k|k}$ is the updated estimate state, $\mathbf{P}_{k|k}$ is the updated estimate covariance.

Equations (17-24) are recursively computed for each new value of (I_x, I_y). After a certain number of iterations, the state or ellipse coefficients $\mathbf{x}$ converges to a constant value, thus the ellipse is optimally estimated and (A, B, D, E, F) are ready to be used for correcting the periodic error.

In fact, the extended Kalman filter is not an optimal estimator due to its linearization strategy, and its result and performance depend on the initial estimate state and the order of input data [33]. There are other advanced but complex EKF, which could achieve a better estimation. However, the general EKF has been shown to work in engineering estimation problems, specifically for ellipse fitting. More importantly, it is practical to implement this algorithm in real-time in hardware.

3.3. Error Correction

In order to correct the periodic error, the correction coefficients I_{xc}, I_{yc}, α, and β are needed in terms of ellipse coefficients (A, B, C, D, E, F) [15],

$$I_{xc} = \frac{2CD - BE}{B^2 - 4AC}, \tag{25}$$

$$I_{yc} = \frac{2AE - BD}{B^2 - 4AC}, \tag{26}$$

$$\alpha = \frac{2A}{\sqrt{4AC - B^2}}, \tag{27}$$

$$\beta = \frac{B}{\sqrt{4AC - B^2}}. \tag{28}$$

With ellipse coefficients approaching to constant values, the correction coefficients also converge eventually, as Figure 4 shows.

Depending on the arctangent implementation, the ways to correct the periodic error may vary. If a single-input arctangent (atan( )) is used, it can

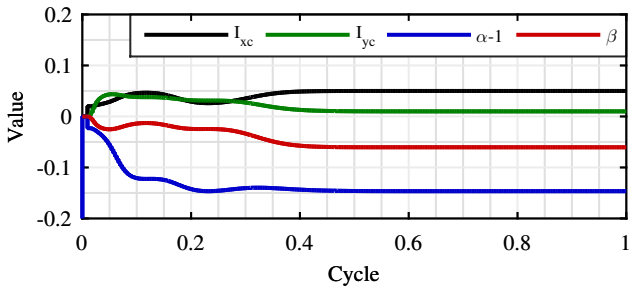


Figure 4: The correction coefficients converge within half of the cycle.

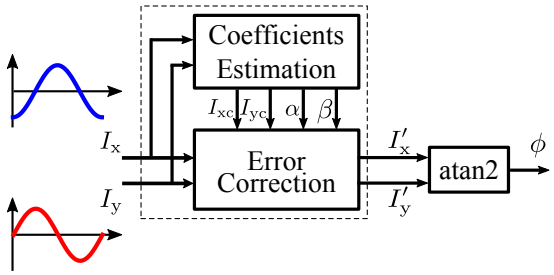


Figure 5: The structure of periodic error correction module. It extracts the correction coefficients from the input in-phase and quadrature signals and applies the coefficients back to the signals for correction.

directly calculate

$$\frac{1}{\frac{I_x - I_{xc}}{I_y - I_{yc}} \cdot \alpha + \beta}, \tag{29}$$

and then feed the result into arctangent (11). If a dual-input arctangent (atan2(,)) is used, it must correct I_x, I_y separately first,

$$I'_x = (I_x - I_{xc}) \cdot \alpha + (I_y - I_{yc}) \cdot \beta, \tag{30}$$

$$I'_y = I_y - I_{yc}, \tag{31}$$

and then feed them into the arctangent separately (10).

4. Implementation

The implementation of this periodic error correction module consists of two parts: one is correction coefficients estimation part, and another is error correction part. The structure is as Figure 5 shows.

As we know, in the middle of the phasemeter process [1, 29], an in-phase signal and a quadrature signal are generated to extract phase by an arctangent. The correction coefficients estimation part uses the in-phase and quadrature

signals to estimate the ellipse coefficients, and then converts them to correction coefficients (25-28). The periodic error correction part is also applied to the in-phase and quadrature signals. By using simple arithmetic operations with the correction coefficients, it corrects the periodic error in in-phase and quadrature signals.

4.1. Correction Coefficients Estimation Design

The correction coefficients estimation part in this design consists of two steps: estimating the ellipse coefficients based on an extended Kalman filter, and then calculating correction coefficients using ellipse coefficients (25-28). The Kalman filter is the critical part in the model. Two solutions are commonly used to implement the Kalman filter in hardware.

One is to program the Kalman filter algorithm in C/C++, and execute it as software in a microprocessor [18, 35], which is straightforward for design and rapid for algorithm verification and prototyping. Due to the occurrence of matrix multiplications, which consist of an amount of basic arithmetic operations, the microprocessor must take a long time to serially execute these operations for each input, which leads to low throughput (thousands of clock cycles per output).

Another is to map the Kalman filter to an HDL-designed, parallel, pipelined architecture in FPGA, also known as systolic array [36]. The systolic array is dedicated hardware for matrix operations specifically. It inherently has higher throughput than the universal arithmetic logic unit (ALU) in a microprocessor, which means it can update those four correction coefficients at a higher frequency. However, the systolic array architecture is difficult to design.

We present a new implementation other than those two previously-reported methods. It implements more dedicated hardware architecture of Kalman filter for the ellipse fitting application specifically, which costs considerably fewer hardware resources and less time to execute.

4.1.1. Microprocessor Based

In this method, the coefficients estimation module is implemented in a microprocessor. An embedded soft processor and a co-processor provide universal ALU and specific floating-point arithmetic operations circuits, which can integrate into an FPGA-based phasemeter [35, 37]. The Kalman filter algorithm for ellipse estimation is implemented in C/C++ software. The software is executed by the microprocessor and manages each fundamental operation of coefficient estimation sequentially.

Due to the limitation of the maximum clock frequency of the soft processor and co-processor, and the complicated matrix operations of the Kalman filter (time complexity: $O(n^2)$), the software cannot process every input (I_x, I_y) for fitting the ellipse. So this downsampling makes it update the correction coefficients in a relatively low frequency. It needs about 1.54×10^4 clock cycles to process one input (I_x, I_y), and the clock frequency is 100 MHz, so this method could only update correction coefficients at 6.5 kHz.

Some further options may increase the updating frequency. For example, if a SoC FPGA is available, the integrated ARM-based hard processor could run up to 800 MHz, which accelerates the processing speed; and it may support fixed-point operations, which is even faster than current floating point operations.

4.1.2. Systolic Array Based

The systolic array is a homogeneous network of Data Processing Units (DPU). Unlike the microprocessor based Kalman filter using the universal ALU in the processor, the systolic array based Kalman filter uses dedicated hardware to achieve matrix operations, which is easy to achieve pipelining and parallel processing. Previous work has proposed to implement the Kalman filter in an FPGA based on a systolic array [36, 38].

The systolic array has advantages in resource efficiency when used for large scale matrix operations, which are high computationally intensive tasks [39], such as image processing, biological sequence comparison, and graph algorithms. However, when the scale of the matrix is very small, the advantages become insignificant. The most complex operation is a 1-by-5 matrix multiplying a 5-by-5 matrix in the ellipse fitting application, which is much lighter than above applications.

4.1.3. Application-specific Architecture Based

This dedicated hardware architecture is for ellipse fitting application specifically. Instead of dealing with matrix operations, this method decomposes the Kalman filter equations to a lower scale, which deals with the single elements. Because the ellipse fitting application only has small scale matrices, the amount of the elements is acceptable.

First, we decompose the Kalman filter equations to several fundamental functions (Table. 1). Some of the fundamental functions depend on the result of previous functions, which means they must wait for the previous functions to be completed. Some functions contain an amount of the same basic operations, which means if they are executed simultaneously, it costs considerable hardware resources. Hence, for those functions depending on the previous one, it is not necessary to assign dedicated hardware for them. They can just share the hardware with the previous one, which optimizes the resource usage and does not influence the overall throughput. For those functions which contain a large number of the same operations, we can make their hardware pipelined and folded, which reuses the hardware but decreases the throughput slightly. Hence, we need to find the most common operation sets, whose hardware could be shared among the functions, but also have balanced parallelism.

Steps 2 and 3, and correction coefficients calculation (25-28) execute similar operation set, which can be abstracted as

$$a + b \times c + d \times e \times f.$$

Its architecture is shown schematically in Figure 6. The hardware of the operation set is designed as pipelined. By sharing the hardware among those steps, it saves the resource and keeps acceptable throughput.

Table 1: Fundamental Functions of Kalman Filter Processing

Step	Function	Op Set
1	$\mathbf{H}(I_x, I_y)$	N/A
2	$\mathbf{R}(I_x, I_y, \mathbf{x})$	1
3	$\mathbf{y}(I_x, I_y, \mathbf{x})$	1
4	$PH_i = \mathbf{P} \times \mathbf{H}^T$	2
5	$HP = \mathbf{H} \times \mathbf{P}$	2
6	$HPH_i = HP \times \mathbf{H}^T$	2
7	$\mathbf{K} = PH_i/(HPH_i + \mathbf{R})$	N/A
8	$\mathbf{x} = \mathbf{x} + \mathbf{K} \times \mathbf{y}$	3
9	$\mathbf{P} = \mathbf{P} - \mathbf{K} \times HP$	3

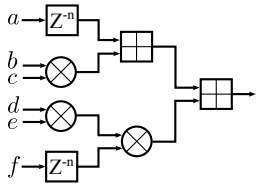


Figure 6: The architecture of operation set 1. And it takes 7 clock cycles from input to result.

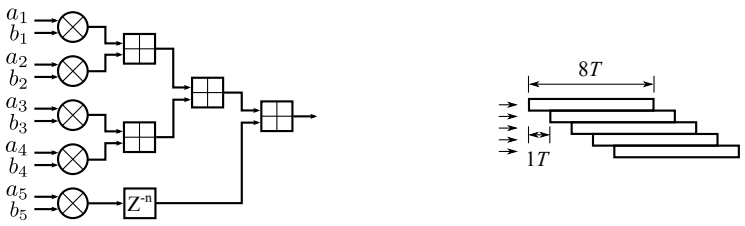
The matrix multiplications in Steps 4, 5, and 6 are $\mathbf{M}_{1,5} \times \mathbf{M}_{5,5}$, $\mathbf{M}_{5,5} \times \mathbf{M}_{5,1}$, and $\mathbf{M}_{1,5} \times \mathbf{M}_{5,1}$[1]. These functions contain a large amount of element addition and multiplication operations. If every addition and multiplication operation was assigned dedicated hardware, it could execute these operations simultaneously and achieve maximum throughput, but cost considerable hardware resources. If they are constrained to share only one adder and one multiplier, it achieves minimum hardware usage, but significantly decreases the throughput resulting from sequential operation. Hence, to balance the hardware usage and throughput, we design a pipelined common operation set:

$$a_1b_1 + a_2b_2 + a_3b_3 + a_4b_4 + a_5b_5,$$

which is between all-dedicated and all-shared hardware. This architecture is shown schematically in Figure 7a.

For instance, the functions $\mathbf{M}_{1,5} \times \mathbf{M}_{5,5}$ needs five of this operation set. As Figure 7b shows, this architecture could feed the operands in at five clock cycles sequentially. Because it is pipelined, it only takes four more clock cycles than all-parallel processing, but save 4/5 of the hardware resource.

[1] $\mathbf{M}_{i,j}$ represents a matrix, which has i rows and j columns.



(a) (b)

Figure 7: (a) is the architecture of operation set 2. Its function is to calculate the summary of five results of multiplication, which is a common operation set in matrix multiplication. The internal signals are exactly synchronous. (b) is the timing for operating this set. It is pipelined, so at each clock it could have a new data in. And it takes 8 clock cycles from input to result.

(a) (b)

Figure 8: (a) is the architecture of operation set 3. Its function is to calculate 6 multiply-accumulate operations in parallel. (b) is the timing for operating this set. At each clock, it feeds the operands for 6 multiply-accumulate operations. And it takes 4 clock cycles from input to result.

Similarly, Steps 8 and 9 also contain large amount of addition/substation and multiplication, but they can be abstracted as multiply-accumulate:

$$a + b \times c.$$

Its architecture is shown in Figure 8a. To calculate $\mathbf{x}$ and $\mathbf{P}$, it needs 30 multiply-accumulate operations in total. Likewise, to balance to resource usage and throughput, we implement an architecture with 6 pipelined multiply-accumulate in parallel, so it only needs to feed the operands in five continuous clock cycles to complete all computation (Figure 8b).

Additionally, with one more standalone adder, multiplier, divider and square root, all nine steps of Kalman filter and correction coefficients calculation (25-28) could be calculated by the combination of above common operation sets and the standalone operations. Meanwhile, it achieves optimal hardware usage and throughput balance.

Eventually, the correction coefficients estimation module in this method needs only 125 9×9 multipliers, which is 23.5% of the multiplier resource on the DE2-115 FPGA board and 9.7% of that on the TR4 FPGA board. DE2-115 and TR4 are two FPGA development boards used in this research.

All of the common operation sets and standalone operations are pipelined. Meanwhile, by carefully arranging the timing of each operation, the overall ellipse coefficients estimation (Kalman filter) and correction coefficients calculation only take 44 clock cycles for one iteration (for easy downsampling, approximate to 50). In this project, the system clock is 50 MHz, so the correction coefficients (α, β, I_{xc}, I_{yc}) can update at 1 MHz.

In aspects of hardware resource usage and throughput, this method is superior to other two methods. It is practical to implement the application-specific architecture based Kalman filter in FPGA.

4.2. Error Correction Design

The error correction part corrects the input (I_x, I_y) based on the correction coefficients. Since the input (I_x, I_y) are fixed-point signals, which are originally digitized by the analog-to-digital converter, and following arctangent operation is also fixed-point, it is necessary to convert the floating-point correction coefficients (α, β, I_{xc}, I_{yc}) to fixed-point first, and then operation fixed-point error correction.

It is difficult and inefficient to implement fixed-point division on an FPGA, so we do not consider single-input atan, which uses a result of division (29) as an operand. A dual-input CORDIC-based atan2 can accept the numerator and denominator (11) separately, which avoids the division operation. Hence, the next step is to calculate the corrected I'_x (30) and I'_y (31) separately.

The fixed-point correction operations can generate an output every clock cycle, so it could correct every input (I_x, I_y) based on the latest correction coefficients.

However, if the periodic error is time-varying, the correction coefficients estimation part needs 50 clock cycles to generate new coefficients, so the latest correction coefficients may not reflect the current state. That is the reason why it is preferential to update the correction coefficients as fast as possible, which make it more accurately represent the current state. However, in this paper, we will only discuss the situation with constant periodic error amplitude.

4.3. Initial Value

Because the Kalman filter is a recursive process, it needs time to approach to optimal estimation. The closer the initial value to the actual value, the sooner it can converge, so the convergence time is important for ellipse fitting. If there is no periodic error in a heterodyne interferometer, the Lissajous curve of (I_x, I_y) is a circle with a radius of 0.5. The ellipse coefficients $\mathbf{x}$ for this circle are $(0.5, 0, 0, 0, -0.125)$. Usually, the ellipse (I_x, I_y) does not deform too much from a circle, it is fair to set it as the initial value for any uncharacterized setup. If the ellipse coefficients are already known from previous measurements,

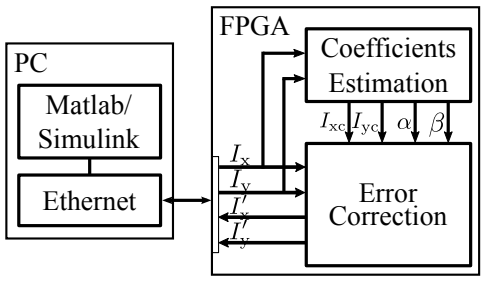


Figure 9: The setup of HIL simulation. On PC side, Stimulus is generated by Matlab/Simulink, and transmitted to FPGA by Ethernet. On FPGA side, the stimulus is processed by periodic error correction module, and the result is transmitted back to PC for further processing and analysis.

the initial value could also be set as those known coefficients, which make it converge sooner. The initial value in practice could be easily customized on demand.

The noise level Σ is another coefficient that can be customized. It determines the convergence process to be rapid but unstable, or gentle but rigid. In this work, we found $\Sigma = 0.05$ to be suitable as the default value.

5. Simulations

To determine the functionality and performance of this periodic error correction module, a series of simulations were performed. By processing computer-generated signals, it isolates other optical and mechanical error sources and obtains information on the error contribution and process capability of this correction module only.

Traditional software simulations for FPGA digital signal processing algorithms have a long simulation runtime, and application-specific availability and accuracy. Because the software cannot fully imitate the conditions and environment of the real world, the results may not reflect its real performance in hardware. A technique called FPGA hardware-in-the-loop (HIL) was used to verify the functionality and performance of the FPGA design. Unlike software simulation, HIL simulation allows data to be processed in real time by the FPGA hardware rather than by the software. The stimulus data are generated by Matlab/Simulink, which could be arbitrary or customized, and fed to the FPGA, and then the FPGA computational results are collected by the Matlab/Simulink for further analysis and display. This approach accelerates simulation time, and also ensures that the algorithm will behave as expected in the real world [40]. The setup of this simulation is shown in Figure 9.

In this paper, two main scenarios were simulated. One is the measured target has a constant velocity, another is the measured target has non-constant velocity, which leads the Doppler frequency to be constant or not. As the Figure 10 shows, when the Doppler frequency is constant, the points on the

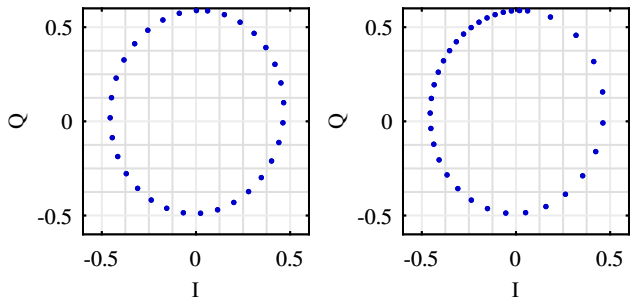


Figure 10: Doppler frequency influences the distribution of samples on Lissajous curve. (a) and (b) show the Lissajous curves of in-phase (I) and quadrature (Q) signals of the measurements with periodic error. Constant Doppler frequency leads even distribution (a), and non-constant Doppler frequency leads uneven distribution (b).

Lissajous curve distribute evenly. However, when the Doppler frequency is non-constant, the point on the Lissajous curve is not evenly. The simulations will figure out whether the distribution of samples impacts the ellipse fitting performance.

In following simulations (5.1, 5.2), it assumes that the amplitude of each order of periodic error is constant. Because generally once interferometer configuration setup, the amplitude of periodic error is almost constant or just slightly changes during one measurement.

5.1. Constant Velocity

Constant velocity is the most common scenario of target motion measurement. Most of the periodic error correction methods deal with this scenario. In this paper, several simulations are performed in different constant velocities to verify its performance in this basic scenario.

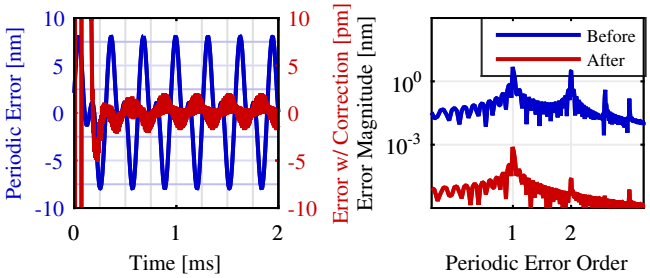
The following simulations simulate the measurements taken by a heterodyne interferometer with 100 kHz split frequency. The velocities of measured target are 1 mm/s (Doppler frequency: 3.16 kHz) and 10 mm/s (Doppler frequency: 31.6 kHz). The sampling frequency of the system is 50 MHz. The computer-generated stimulus are given by:

$$I_x = 0.5\left((1+\gamma_{21})\cos\left(2\pi f_d\right) - \gamma_{22}\sin\left(2\pi f_d\right) + \gamma_{11}\right), \tag{32}$$

$$I_y = 0.5\left((\gamma_{21}-1)\sin\left(2\pi f_d\right) + \gamma_{22}\cos\left(2\pi f_d\right) + \gamma_{12}\right), \tag{33}$$

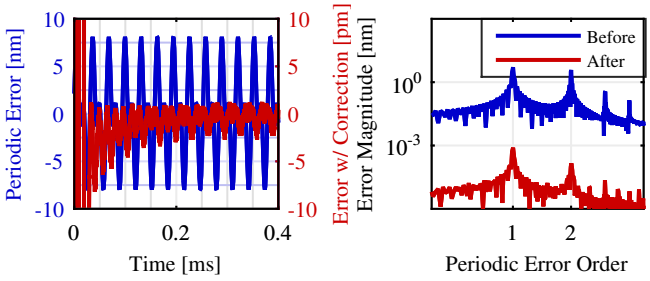
where f_d is Doppler frequency, which is a constant value. Coefficients γ_{11} is 0.1, γ_{12} is 0.02, γ_{21} is 0.08, γ_{22} is 0.03.

From Figure 11a and Figure 12a, we can see the residual error (red) takes a short time to reach steady state, and then keeps at picometer levels. In frequency domain, for 1 mm/s velocity simulation (Figure 11b), the periodic error correction method could decrease first- and second- order error by 75.9 dB



(a) (b)

Figure 11: Comparison of periodic error before and after correction. The Doppler frequency is 3.16 kHz. In the time domain (a), the periodic error attenuates from ±8.0 nm to ±2.1 pm, RMS attenuates from 4.7 nm to 0.7 pm. In the frequency domain (b), the first- and second-order error attenuate 75.9 dB and 102.0 dB, respectively.



(a) (b)

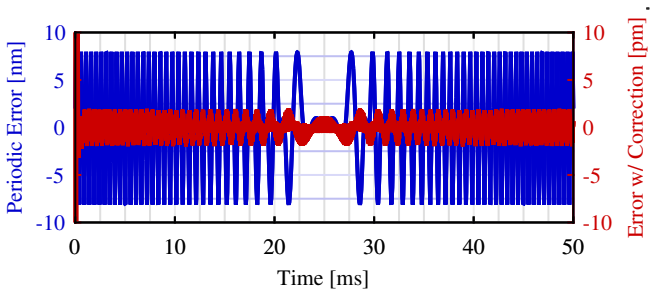
Figure 12: Comparison of periodic error before and after correction. The Doppler frequency is 31.6 kHz. In the time domain (a), the periodic error attenuates from ±8.0 nm to ±2.1 pm, RMS attenuates from 4.7 nm to 0.7 pm. In the frequency domain (b), the first- and second-order error attenuate 76.2 dB and 88.4 dB, respectively.

and 102.0 dB; for 10 mm/s velocity simulation (Figure 12b), it could decrease first- and second- order error by 76.2 dB and 88.4 dB.

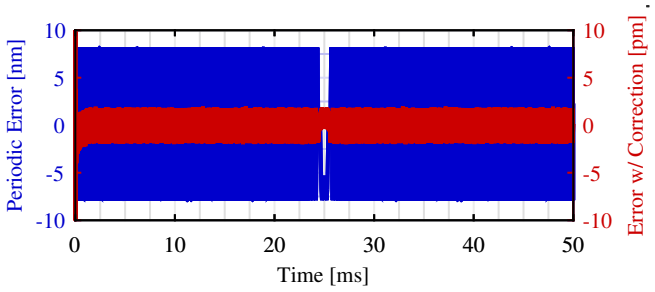
According to the simulation results, the residual errors after correction are just slightly varying among different Doppler frequencies, but significantly attenuated compared with original periodic errors. So this implementation can effectively correct periodic error for constant velocity measurements, and has very high correction performance consistently.

5.2. Non-constant Velocity

Non-constant velocity is a common scenario for target motion measurement as well, including changing direction, accelerating, and decelerating. The following simulations simulate the target moving in sinusoidal velocities, which is typical motion in manufacturing processes.



(a)



(b)

Figure 13: Comparison of periodic error before and after correction, when the Doppler frequency is oscillating at 100 Hz. When the peak Doppler frequency is 1.99 kHz (0.63 mm/s) (a), the periodic error attenuates from ±8.0 nm to ±2.1 pm, RMS attenuates from 4.5 nm to 0.6 pm. When the peak Doppler frequency is 49.6 kHz (15.7 mm/s) (b), the periodic error attenuates from ±8.0 nm to ±2.3 pm, RMS attenuates from 4.7 nm to 0.7 pm.

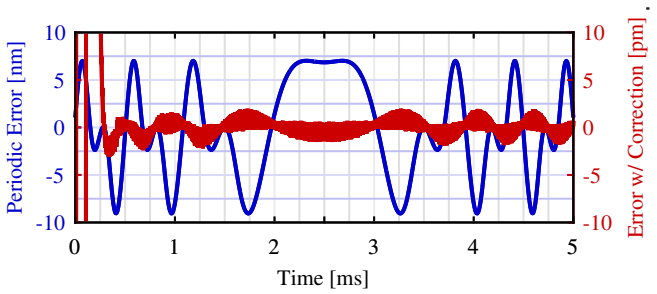
Four simulations are performed. They are the combination of two peak velocities 0.63 mm/s and 15.7 mm/s, and two frequencies of oscillating 10 Hz and 100 Hz. The other parameters are the same as that in the constant velocity simulation. The computer-generated stimulus are similar to (32,33), except the Doppler frequency is non-constant:

$$f_d = f_p \sin\left(2\pi f_o t\right), \tag{34}$$

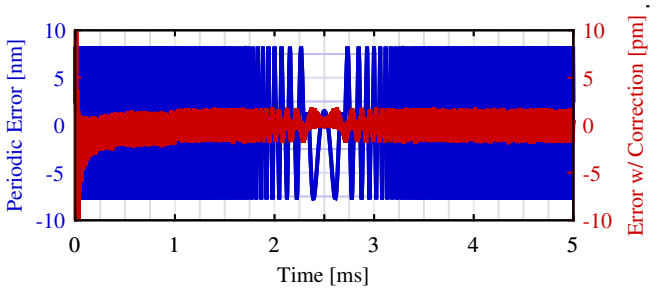
where f_p is peak Doppler frequency, f_o is the frequency of velocity oscillating.

From Figure 13 and Figure 14, no matter what the peak velocity (Doppler frequency) or the frequency of oscillating is, the peak-to-peak periodic error and RMS of periodic error decrease to picometer level, which is more than three orders of magnitude lower than the original errors. Because the Doppler frequency is continuously changing during one measurement, only time-domain plots of periodic error are given here.

According to the simulation results, the amplitude of residual error after correction are the almost same for these four simulations, and significantly decreased compared with the original periodic errors. So the uneven distribution of samples does not influence the accuracy of ellipse fitting, and the error



(a)



(b)

Figure 14: Comparison of periodic error before and after correction, when the Doppler frequency is oscillating at 10 Hz. When the peak Doppler frequency is 1.99 kHz (0.63 mm/s) (a), the periodic error attenuates from ±8.0 nm to ±2.2 pm, RMS attenuates from 5.2 nm to 0.6 pm. When the peak Doppler frequency is 49.6 kHz (15.7 mm/s) (b), the periodic error attenuates from ±8.0 nm to ±2.1 pm, RMS attenuates from 4.6 nm to 0.7 pm.

correction performance for non-constant velocity is comparable to that for constant velocity.

5.3. Performances

All previous simulations test the functionality, however, the performances, like the speed of convergence and noise tolerance, are also critical specifications [16, 30]. Hence, we also investigate the speed of convergence and the RMS of residual noise in output for the different input noise levels.

Mathematically, the noise level is Σ in (18). The higher noise level leads to the higher covariance of observation noise $\mathbf{R}_k$, thus, leads to smaller optimal Kalman gain $\mathbf{K}_k$ in (22). According to (19) and (23),

$$\mathbf{x}_{k|k} = \mathbf{x}_{0|0} + \sum_{i=0}^{k} \mathbf{K}_i \mathbf{y}_i, \tag{35}$$

where $\mathbf{x}_{k|k}$ is the updated estimate state at k-th iteration, $\mathbf{x}_{0|0}$ is the initial state. Assume $\mathbf{x}$ is the real state, when $\mathbf{x}_{k|k}$ is within $\mathbf{x} \pm \varepsilon$, we consider it as

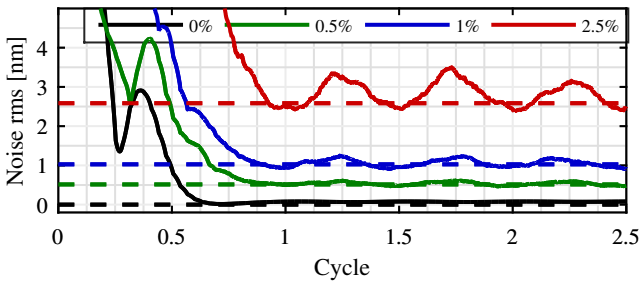


Figure 15: The RMS of the residual noise in output when adding different levels of noise to input signals. The solid lines are the local RMS of 50 points of residual noise after correction. Using local RMS instead of global RMS eliminates the influence of transient state at the beginning. The dash lines are the levels of input noise. Because the Kalman filter in this design only deals with periodic error, the white noise in the inputs will pass through and be left in the output.

the real state. Thus,

$$\left|\mathbf{x}-\mathbf{x}_{k|k}\right| = \left|\mathbf{x}-\mathbf{x}_{0|0}-\sum_{i=0}^{k}\mathbf{K}_i\mathbf{y}_i\right| \leq \varepsilon. \tag{36}$$

If $\mathbf{K}_i$ becomes smaller for each iteration, it needs more iterations to hold inequality (36), which slows the speed of convergence.

To test its noise tolerance and convergence speed, we add 0.5%, 1% and 2.5% white noise to the input signals[2], and then compare the RMS of noise in output with and without correction. The stimulus signals I_x, I_y are generated based on the correction coefficients (α, β, I_{xc}, I_{yc}), which are set as $(1.1434, -0.1676, -0.06854, -0.0702)$ in this test. The results are shown in Figure 15.

From the simulations, we can see the white noise introduced to input signals also appears in the output, because the periodic error correction does not have a mechanism to eliminate the noise. Meanwhile, the residual noise levels approach to input noise levels eventually, which means it will not amplify the noise as well. However, the noises do impact the performance of the correction. The large ripple in the red line reflects that variations caused by noise on input I_x, I_y impact the accuracy of the ellipse fitting, and the higher level noise impacts the fitting much more significant. The noise also influences the speed of convergence. In the ideal case (0% noise), it needs only 0.6 cycle to achieve the optimal estimation, however, along with the noise increasing to 2.5%, it needs about 1 cycle but still not stable, which means that it may need 633 nm (1 cycle) length to get stable correction coefficients in real measurement.

[2]In fact, the profiles of actual noise in I_x, I_y for homodyne and heterodyne interferometers may be different. For heterodyne interferometer, I_x, I_y are filtered by previous low-pass filters, so the spectrum of the noise should decrease towards high frequency, depends on the characteristic of the filters. For simplification, we use white noise for simulations.

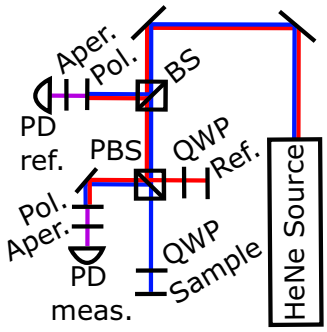


Figure 16: The optical layout of the heterodyne interferometer used for testing. Apertures (aper.) are used to set the level of periodic error.

Several techniques can be applied to front-end digital signal processing system, photodetector system, and optical setup to attenuate the noise, or increase signal-to-noise ratio, such as using filters to attenuate particular frequency band of noise, isolating the optical setup and detectors to avoid the light and temperature variation from the environment. Controlling the noise level of the input signals is helpful to achieve optimal periodic error correction performance.

6. Measurement Validation

We also use real measurement data to validate our implementation. A heterodyne Twyman-Green interferometer was used to measure the displacement of the nPoint nano-positioning stage, the interferometer's optical layout is shown in Figure 16. The interferometer is mounted on an optical isolation table to minimize vibration noise. A Zeeman laser (Hewlett Packard, HP 5518A) with a split frequency of 1.91 MHz was used as a source for the heterodyne interferometer. The beam passes through a non-polarizing beamsplitter (BS), where the reflected beam is interfered using a polarizer and detected on a photodiode (PD) generating the optical reference. The other portion of the beam travels to a polarizing beam splitter (PBS). Here, the two orthogonal beams split via polarization and frequency. Each passes through a quarter wave plate (QWP), before reflecting from the reference and measurement mirrors. Upon reflection, the beams pass back through the QWP, rotating their respective polarizations a total of 90°. Now, the originally reflected beam transmits through the PBS and vice versa for the originally transmitted beam. The two beams are then interfered with a polarizer and detected using a PD to generate the measurement signal. Apertures are used on the reference and measurement beams directly before the PDs to help set the level of periodic error.

The heterodyne frequency of the laser is 1.91 MHz, which is higher than the maximum bandwidth of 100 kHz of the lock-in amplifier (Sanford Research,

SR830) used to measure the phase change. This was solved by down mixing the 1.91 MHz heterodyne frequency with a common local oscillator driven at 1.88 MHz using a function generator (Tektronix, AFG3022B) and frequency mixers (Mini-Circuits, AZD-1-1+). The signal output from the frequency mixers has both 30 kHz and 3.79 MHz components. The higher frequency was removed using a low pass filter (Krohn-Hite, 3944) to precondition the signal for the lock-in amplifier, and amplify the signals by 20 dB. The lock-in amplifier produces in-phase (8) and quadrature (9) signals for following phase measurement and error correction use. The data from the lock-in amplifier was recorded with LabView and an NI-6529 DAQ card on a Windows PC. The NI DAQ was also used to generate voltage drive signals, which were directly sent to the nPoint controller. The built-in capacitive sensor of nPoint stage provides its instantaneous position under closed-loop control. The readout of the capacitive sensor is used as the reference to examine the periodic error in interferometer measurement.

We met a technical difficulty in recording capacitive sensor readout and periodic error correction module readout simultaneously. For comparison purpose, we chose simultaneously record capacitive sensor readout, in-phase, and quadrature signals instead, and then feed the in-phase and quadrature signals to correction module.

Several constant and non-constant velocity measurement profiles have been taken. Figure 17 shows two constant velocity motion measurements comparison among capacitive sensor, interferometer, and interferometer with error correction.

When the target has a constant velocity, its displacement is a linear ramp ideally. Figure 17a and 17c show the residual errors after removing the linear trend of the displacement. The raw DMI measurement has an error about ± 20 nm, which is a superposition of periodic error, the effects of temperature and refractive index drift, utility frequency (60 Hz and its harmonics), and data acquisition system. After periodic error correction, the residual errors go down to ± 2 nm, which is also smaller than the noise of capacitive sensor measurement ± 6 nm. Figure 17b and 17d show the spectrum of residual errors. The first and second order periodic errors are removed after correction, whose amplitudes are attenuated about 50 dB and 30 dB, respectively. There is a third order periodic error that appears in the spectrum due to misalignment, but it is attenuated to the baseline of noise too. Some other peaks appear at the same position on the three spectrums. They are the common residual error or noise from utility frequency, environment, data acquisition system, and the stage itself.

Two non-constant velocity measurement profiles also have been measured to compare the residual errors among capacitive sensor, interferometer, and interferometer with error correction. In this case, the stage has sinusoidal velocity, sinusoidal displacement, and its periodic error has a time-varying frequency. Figure 18 shows the errors of each measurement. There is a common drift on the errors, which is caused by the respond of stage's closed-loop control. Regardless of the drift, the capacitive sensor has a noise about ± 6 nm. The raw DMI measurement has an error about ± 20 nm, which has the same sources as the previous measurement. After residual correction, the error goes down to

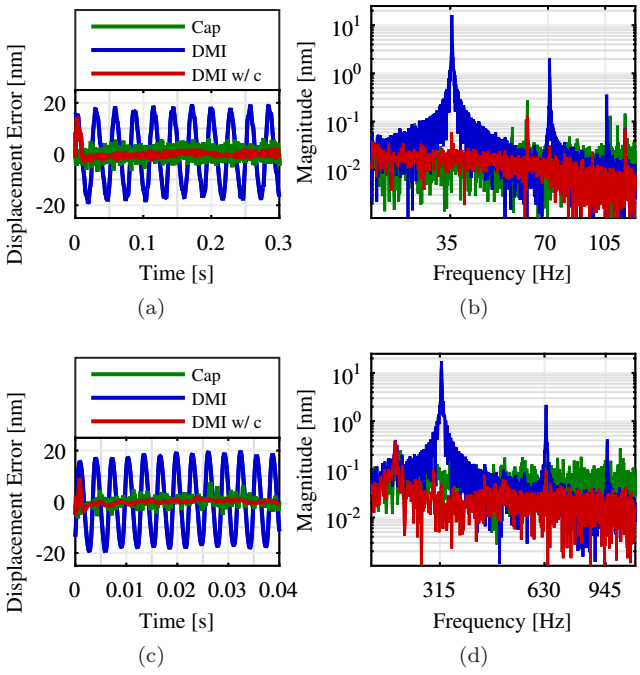


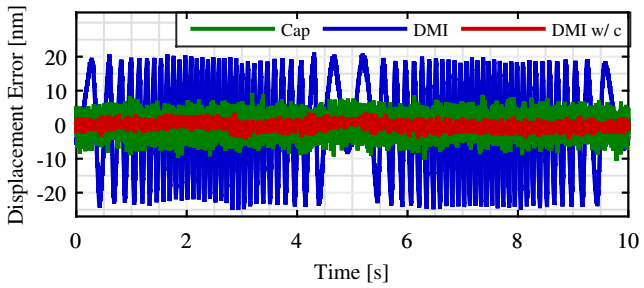
Figure 17: Measurement error comparison among capacitive sensor, interferometer, and interferometer with error correction. (a) and (b) are the residual error and its spectrum for 11μm/s motion (Doppler frequency 35Hz) measurement. After correction, the first- and second- order error attenuate 48.5 dB and 36.5 dB, respectively. (c) and (d) are the residual error and its spectrum for 99.7μm/s motion (Doppler frequency 315Hz) measurement. After correction, the first- and second- order error attenuate 51.1 dB and 26.9 dB, respectively.

±1.5 nm. Those residual errors have the same amplitudes as that in constant velocity scenario. So this correction module has consistent performance in both constant and non-constant velocity measurements.

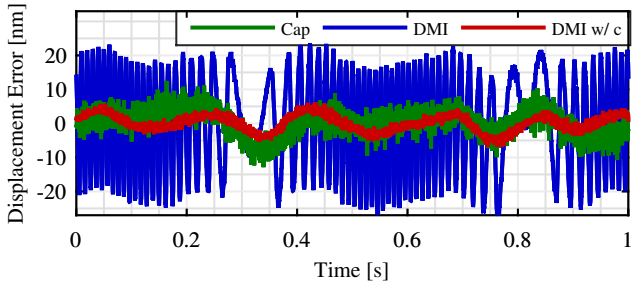
Above measurements are just a part of the measurements we did, all of them show that Kalman filter algorithm and implementation could attenuate the periodic error effectively, no matter what the velocity is and how the velocity changes. Currently, the residual error after correction is about ±1.5 nm, which are all from other sources. The periodic error is effectively corrected.

7. Conclusion

We present a periodic error correction implementation for the heterodyne interferometer, and compatible with the homodyne interferometer. This module can on-line estimate and update correction coefficients at 1 MHz, and real-time correct the periodic error at 50 MHz. It also has the capability to deal with slow time-varying periodic error.



(a)



(b)

Figure 18: Measurement error comparison among capacitive sensor, interferometer, and interferometer with error correction. (a) illustrates the residual errors when measuring a sinusoidal motion: $5\mu\text{m}\cdot\sin(2\pi\cdot 0.1\text{Hz}\cdot t)$. (b) illustrates the residual errors when measuring a sinusoidal motion: $5\mu\text{m}\cdot\sin(2\pi\cdot 1\text{Hz}\cdot t)$.

This periodic error correction module is based on ellipse fitting using extended Kalman filter. Its application-specific hardware architecture in FPGA takes only 44 clock cycles to finish one iteration of the Kalman filter process. This implementation has advantages in throughput and resource usage compared with conventional implementations.

In the measurement validation, the module can effectively eliminate the periodic error for both constant and non-constant velocities. The residual error after correction is about ± 1.5 nm, which is background noise limited by the DMI system, not the periodic correction module.

This technique needs approximately one cycle (I_x, I_y) data to get stable correction coefficients, which means it needs to move 633 nm to estimate the coefficients. For the displacement measurement shorter than 633 nm, it cannot estimate stable coefficients during measurement, so a prior estimating process has to be performed before measurement.

In the future, we will build a better interferometer setup to achieve lower background noise. We will also design an interferometer configuration, which could introduce time-varying periodic error, and test the correction performance of this module for the time-varying periodic error.

Acknowledgment

This work was supported in part by the US Department of Commerce, National Institute of Standards and Technology (NIST) under Award No. 70NANB14H262 and the National Science Foundation under Awards CMMI:1265824. E.D. Burnham-Fay is supported by the Laboratory for Laser Energetics' Horton Fellowship.

References

[1] J. D. Ellis, Field Guide to Displacement Measuring Interferometry, SPIE Press, Bellingham, Washington USA, 2013.

[2] V. Badami, P. de Groot, Displacement measuring interferometry, Handbook of Optical Dimensional Metrology (2013) 157–238.

[3] Methods for performance evaluation of computer numerically controlled (cnc) lathes and turning centers (1998).

[4] Methods for performance evaluation of computer numerically controlled machining centers (2005).

[5] Test code for machine tools - part 1: Geometric accuracy of machines operating under no-load or finishing conditions (2012).

[6] R. R. Fesperman, N. Brown, K. Elliott, J. D. Ellis, A. Grabowski, S. Ludwick, S. Maneuf, B. O'Connor, S. Woody, Methods for performance evaluation of single axis positioning systems: a new standard, in: Proceedings of the 28th ASPE Annual Meeting, Vol. 56, Saint Paul, MN, USA, 2013, pp. 498–503.

[7] A. Rosenbluth, N. Bobroff, Optical sources of non-linearity in heterodyne interferometers, Precision Engineering 12 (1) (1990) 7–11.

[8] N. Bobroff, Recent advances in displacement measuring interferometry, Measurement Science and Technology 4 (9) (1993) 907.

[9] R. C. Quenelle, Nonlinearity in interferometer measurements, Hewlett-Packard Journal 34 (1983) 10.

[10] M. Tanaka, T. Yamagami, K. Nakayama, Linear interpolation of periodic error in a heterodyne laser interferometer at subnanometer levels [dimension measurement], Instrumentation and Measurement, IEEE Transactions on 38 (2) (1989) 552–554.

[11] C.-M. Wu, J. Lawall, R. D. Deslattes, Heterodyne interferometer with subatomic periodic nonlinearity, Appl. Opt. 38 (19) (1999) 4089–4094.

[12] W. Schluchter, R. Belt, Low non-linear error displacement measuring interferometer, uS Patent 7,251,039 (Jul. 31 2007).

[13] K.-N. Joo, J. D. Ellis, E. S. Buice, J. W. Spronck, R. H. M. Schmidt, High resolution heterodyne interferometer without detectable periodic nonlinearity, Opt. Express 18 (2) (2010) 1159–1165.

[14] J. D. Ellis, A. J. H. Meskers, J. W. Spronck, R. H. M. Schmidt, Fiber-coupled displacement interferometry without periodic nonlinearity, Opt. Lett. 36 (18) (2011) 3584–3586.

[15] P. L. M. Heydemann, Determination and correction of quadrature fringe measurement errors in interferometers, Appl. Opt. 20 (19) (1981) 3382–3384.

[16] K. Birch, Optical fringe subdivision with nanometric accuracy, Precision Engineering 12 (4) (1990) 195–198.

[17] C.-M. Wu, C.-S. Su, G.-S. Peng, Correction of nonlinearity in one-frequency optical interferometry, Measurement Science and Technology 7 (4) (1996) 520.

[18] C. H. Wang, A. T. Augousti, J. Mason, Real time evaluation and correction of nonlinear errors in single frequency interferometers, Transactions of the Institute of Measurement and Control 22 (5) (2000) 405–412. doi:10.1177/014233120002200505.

[19] T. Eom, T. Choi, K. Lee, H. Choi, S. Lee, A simple method for the compensation of the nonlinearity in the heterodyne interferometer, Measurement Science and Technology 13 (2) (2002) 222.

[20] T. B. Eom, J. A. Kim, C.-S. Kang, B. C. Park, J. W. Kim, A simple phase-encoding electronics for reducing the nonlinearity error of a heterodyne interferometer, Measurement Science and Technology 19 (7) (2008) 075302.

[21] J.-A. Kim, J. W. Kim, C.-S. Kang, T. B. Eom, J. Ahn, A digital signal processing module for real-time compensation of nonlinearity in a homodyne interferometer using a field-programmable gate array, Measurement Science and Technology 20 (1) (2009) 017003.

[22] T. L. Schmitz, D. C. Chu, H. S. Kim, First and second order periodic error measurement for non-constant velocity motions, Precision Engineering 33 (4) (2009) 353 – 361.

[23] P. Chawah, A. Sourice, G. Plantier, J. Chery, Real time and adaptive kalman filter for joint nanometric displacement estimation, parameters tracking and drift correction of effpi sensor systems, in: Sensors, 2011 IEEE, 2011, pp. 882–885.

[24] P. Hu, J. Zhu, X. Guo, J. Tan, Compensation for the variable cyclic error in homodyne laser interferometers, Sensors 15 (2) (2015) 3090.

[25] C.-M. Wu, R. D. Deslattes, Analytical modeling of the periodic nonlinearity in heterodyne interferometry, Appl. Opt. 37 (28) (1998) 6696–6700.

[26] V. G. Badami, Investigation and compensation of periodic nonlinearities in heterodyne interferometry, Ph.D. thesis, THE UNIVERSITY OF NORTH CAROLINA AT CHARLOTTE (1999).

[27] V. Badami, S. Patterson, A frequency domain method for the measurement of nonlinearity in heterodyne interferometry, Precision Engineering 24 (1) (2000) 41 – 49.

[28] S. Cosijns, H. Haitjema, P. Schellekens, Modeling and verifying non-linearities in heterodyne displacement interferometry, Precision Engineering 26 (4) (2002) 448 – 455.

[29] C. Wang, FPGA-based, 4-channel, high-speed phasemeter for heterodyne interferometry, Master's thesis, University of Rochester, Rochester, NY, USA (2013).

[30] Z. Li, K. Herrmann, F. Pohlenz, A neural network approach to correcting nonlinearity in optical interferometers, Measurement Science and Technology 14 (3) (2003) 376.

[31] C. Lu, J. R. Troutman, J. D. Ellis, T. L. Schmitz, J. A. Tarbutton, Periodic error compensation using the continuous wavelet transform, in: Proceedings of the 29th ASPE Annual Meeting, Boston, MA, USA, 2014, pp. 408–411.

[32] J. Porrill, Fitting ellipses and predicting confidence envelopes using a bias corrected kalman filter, Image Vision Comput. 8 (1) (1990) 37–41.

[33] Z. Zhang, Parameter estimation techniques: a tutorial with application to conic fitting, Image and Vision Computing 15 (1) (1997) 59–76.

[34] R. E. Kalman, A New Approach to Linear Filtering and Prediction Problems, Transactions of the ASME - Journal of Basic Engineering (82 (Series D)) (1960) 35–45.

[35] C. Wang, J. D. Ellis, Real-time periodic error correction for heterodyne interferometers using kalman filters, in: Proceedings of the 30th ASPE Annual Meeting, Austin, TX, USA, 2015.

[36] A. Bigdeli, M. Biglari-Abhari, Z. Salcic, Y. T. Lai, A new pipelined systolic array-based architecture for matrix inversion in fpgas with kalman filter case study, EURASIP J. Appl. Signal Process. 2006 (2006) 75–75.

[37] C. Wang, J. Ellis, Dynamic doppler frequency shift errors: Measurement, characterization, and compensation, Instrumentation and Measurement, IEEE Transactions on 64 (7) (2015) 1994–2004.

[38] A. Sudarsanam, Analysis of field programmable gate array-based kalman filter architectures, Ph.D. thesis, Utah State University (2010).

[39] K. Johnson, A. Hurson, B. Shirazi, General-purpose systolic arrays, Computer 26 (11) (1993) 20–31.

[40] Hardware in the loop from the matlab/simulink environment, White Paper (Sep. 2013).