ENHANCED STEP MODE FTIR POSITION CONTROL

by

R. Brandon Inberg

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Electrical and Computer Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

August 2005

© Copyright

by

R. Brandon Inberg

2005

All Rights Reserved

APPROVAL

of a thesis submitted by

R. Brandon Inberg

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

Dr. Steven R. Shaw

Approved for the Department of Electrical and Computer Engineering

Dr. James Peterson

Approved for the College of Graduate Studies

Dr. Joseph J. Fedock

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library.

If I have indicated my intention to copyright this thesis by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extend quotation from or reproduction of this thesis in whole or in parts may be granted only by the copyright holder.

R. Brandon Inberg

August 2005

ACKNOWLEDGEMENTS

I would like to thank Dr. Steve Shaw for all of his advice and guidance through out this project. I greatly appreciate all of the time he spent helping me out and all of his great ideas. I would also like to thank Dr. Lee Spangler and his group, for helping me use the FTIR spectrometer.

TABLE OF CONTENTS

1	INTRODUCTION	1
	FTIR Spectroscopy	3
	Nyquist Sampling Limitation	5
	FTIR Modes of Operation	6
	Continuous Scan Mode	6
	Step-Scan Mode	6
	Time Resolved Spectroscopy Experiments	7
	Rapid and Ultrarapid TRS Measurements	7
	Stroboscopic TRS Measurements	8
	Step-Scan TRS Measurements	9
	Overview of Proposed Position Control	9
2	APPROACH AND METHOD	11
	Feedback Dynamics	12
	Approach For Enhanced Position Control	14
	Modeling and Estimation	16
	Properties of The Estimate	21
3	IMPLEMENTATION	24
	Hardware	24
	Software	26
	Software Implementation	29
	Operational Procedure and Experimental Setup	31
4	EXPERIMENTAL PROCEDURE AND RESULTS	34
	Closed Loop Mirror Positions	34
	DSP Mirror Estimate	35
	Spectral Experiment	36
5	CONCLUSION	40
	REFERENCES	42
	APPENDICES	43
	APPENDIX A : PCB SCHEMATIC	44
	APPENDIX B : DSP CODE	47
	APPENDIX C : MATLAB CODE	68

LIST OF FIGURES

Figure		Page
1	Conceptual diagram of a Michelson interferometer.	3
2	Flow diagram of transmission FTIR. Sample and background interferograms are taken. This data is FFTed to produce a spectrogram. A transmittance diagram of the sample is generated by dividing the sample spectrogram by the background spectrogram.	5
3	Schematic diagram of mirror position control system in the Brüker FTIR spectrometer.	12
4	FTIR mirror position control as a feedback system.	13
5	Measured and simulated detector output waveforms. Simulated detector outputs are shifted to correspond with the oscilloscope display . .	15
6	FTIR mirror position control modified for finer spatial resolution. . .	16
7	This represents the mirror parameterization from (17). The idea is to estimate b_1 , b_2 , r , and ϕ at the same time.	18
8	This shows how the detector signals y_1 and y_2 are mapped to q_1 and q_2 . .	19
9	A software flow diagram for the DSP.	21
10	Histogram of 20,000 bootstrap estimates for the residual $(\phi - \hat{\phi})$ with $SNR = 67$. The mean value for the outer distribution is $-5.3e - 04$ and the mean for the inner distribution is $-3.1e - 05$	22
11	A picture of the DSP module used to estimate the mirror position and move the mirror on a finer grid.	25
12	A schematic of the interface circuitry for the DSP module.	26
13	Amplitude A and temperature change as a function of time.	28
14	This figure shows the ideal stopping mirror positions along with the actual stopping positions. The cross arrows show the ideal mirror positions and the clusters are where the mirror actually stopped. The dotted line is the ideal mirror path.	35
15	A block diagram for a test that was done to check if the DSP module was producing biased mirror positions. The DSP module was tested against an ideal DSP model.	36

LIST OF FIGURES – CONTINUED

Figure		Page
16	Measured DSP output and model output in 3x mode. The empirical distribution of errors is very close to zero mean.	37
17	The lower section of this graph contains four plot lines showing the background spectrograms (the two upper lines) and the erbium crystal sample spectrograms (two lower lines). In the upper section of this graph, the transmittance spectrum of the erbium crystal sample is shown as determined in step and continuous scan modes. The step-scan data was collected in N=3 mode and the continuous scan data was collected utilizing the PLL with the DSP in “pass through” mode.	39

ABSTRACT

This thesis presents a modification to a typical Fourier transform infrared (FTIR) spectrometer to achieve finer spatial sampling and increased frequency range for step-scan experiments with little modification to the existing hardware. Commercially available step-scan FTIR spectrometers currently have the ability to measure spectra up to 15798 cm^{-1} , which is limited by a HeNe reference laser used for controlling the mirror's position. The proposed technique involves using a digital signal processor (DSP) to measure the HeNe reference signal and estimate the mirror position. The DSP then outputs a new synthetic signal that is used by the spectrometer to control the mirror to finer steps, allowing it to measure spectrums up to 47394 cm^{-1} . Enhanced closed-loop mirror position data is presented to show the quality of the DSP estimate along with spectrograms taken of an erbium crystal to show the overall spectral improvements. To validate the erbium crystal enhanced step-scan spectrogram, a continuous scan experiment is given for comparison. The results demonstrate previously unattainable step-scan performance.

INTRODUCTION

Infrared spectroscopy can be used to explore the chemical properties of a sample, including chemical bonds, molecular geometries, molecular energy levels, and molecular interactions. The technique can be used to determine a sample's static properties as well as providing time resolved information for dynamic events that occur in a sample. In a standard experiment, light with a broad range of frequencies is directed through a sample while the transmittance of the light through the sample is measured. The transmittance spectrum of the sample is found by dividing the transmittance spectrum of the sample by the spectrum of the light source.

Performing infrared spectroscopy with a steady state light source can provide analytical information on a sample with regard to its chemical composition and respective concentrations. By using a pulsed light source, kinetic events can be monitored, giving additional insight into a sample's chemical makeup. Experiments can be performed to gain information regarding a sample's atomic geometry and molecular interactions. These types of experiments are categorized as time resolved spectroscopy (TRS).

There are two popular techniques for doing infrared spectroscopy [7]. The first method uses a diffraction grating to spatially separate the spectrum. The frequency resolution of this spectrometer is a function of the fineness of the grating. A scanning mirror can be used to channel one wavelength of light into the detector. The second method involves using an interferometer to encode the spectrum in the measured signal. This method is called Fourier transform infrared (FTIR) spectroscopy. A FTIR spectrometer uses a Michelson interferometer to modulate the optical signal and encode the spectrum information. A Fourier transform is performed on the resulting signal to retrieve the frequency/magnitude information. The upper limit of

the instrument's optical frequency measurement spectrum is limited by its ability to accurately resolve the position of its scanning mirror. In other words, by allowing the instrument to measure the position of the scanning mirror in finer increments, the instrument is able to measure optical signals with shorter wavelengths, thus increasing its optical spectrum measurement range. Also, if the scanning mirror's distance of travel is increased, the instrument's resolution over its optical frequency spectrum is increased.

There are three main advantages of using a FTIR spectrometer over dispersive spectrometers [4]. The first advantage is all of the wavelengths are measured simultaneously with one detector. This allows the FTIR spectrometer to produce a spectrum very quickly so it is able to combine multiple scans for a higher signal-to-noise ratio as compared to scanning dispersive spectrometers, where in the same time it will only be able to do one scan. This advantage is known as the multiplex or Fellgett advantage. The next advantage is that more signal reaches the detector as compared to a dispersive system. The FTIR spectrometer does not separate the signal energy into individual frequencies as a dispersive system does. This means that for higher resolution measurements, there is less energy at the detector for a dispersive spectrometer as compared to the FTIR. The amount of energy is constant with respect to frequency resolution for an FTIR spectrometer. This is known as the throughput or Jacquinot advantage. The last advantage is the high resolution and precision the FTIR spectrometer provides. The resolution and precision come from the ability to move the mirror over long distances and have good mirror alignment while being able to measure the mirror's position very precisely with the use of a stable HeNe laser for the mirror position feedback. These are the reasons that the FTIR is the most common mid-infrared spectrometer used today.

Figure 1: Conceptual diagram of a Michelson interferometer.

FTIR Spectroscopy

The key component in a FTIR spectrometer is the Michelson interferometer. Light with intensity G is the input to the interferometer consisting of a beam-splitter, two mirrors, and a detector, as shown in Fig. 1. One mirror is a fixed distance from the beam splitter, while the other mirror is able to move. The moving mirror allows an adjustable optical path length difference p , also known as the retardation, so that the two paths of light interfere when they recombine at the beam splitter. The interference can be constructive or destructive depending on p and the wavenumber ν . When the optical path length difference p is equal to zero, all of the frequencies are constructively interfering. The output intensity $I(p, \nu)$ at wavenumber ν is [1]

$$I(p, \nu) = G(\nu)(1 + \cos(2\pi\nu p)). \quad (1)$$

The detector effectively sums the intensity components over the range of frequencies present in the interferometer output to provide a response

$$I(p) = \int_0^\infty I(p, \nu) d\nu = \int_0^\infty G(\nu)(1 + \cos(2\pi\nu p)) d\nu. \quad (2)$$

Rearranging (2) into its DC and AC components gives

$$I(p) = \overbrace{\int_0^\infty G(\nu) d\nu}^{DC} + \overbrace{\int_0^\infty G(\nu)(\cos(2\pi\nu p)) d\nu}^{AC}. \quad (3)$$

The detector's output is AC coupled producing an electronic signal

$$I_d(p) = \int_0^\infty G(\nu)(\cos(2\pi\nu p)) d\nu, \quad (4)$$

which is the real part of an inverse Fourier transform. The spectrum $G(\nu)$ of the sample can be recovered from the measurements $I_d(p)$ by taking the Fourier transform. In a practical instrument, measurements $I_d(p)$ are digitized for a set of positions and the spectrum $G(\nu)$ is determined by FFT.

For a typical experiment, a background interferogram is generated for the light source with an intensity $G(\nu)$ and then again, with the same setup, for a sample placed either before or after the interferometer, producing an interferogram response of

$$I_s(p) = \int_0^\infty G(\nu)S(\nu)(1 + \cos(2\pi\nu p)) d\nu. \quad (5)$$

The intensity of light after being transmitted through the sample is $G(\nu)S(\nu)$, where $S(\nu)$ is the transmittance of the sample. The background and sample interferogram's are FFTed to produce spectrogram's $G(\nu)$ and $G(\nu)S(\nu)$, respectively. The sample's transmittance spectrum is found by dividing the sample spectrum by the background spectrum, i.e.

$$S(\nu) = \frac{G(\nu)S(\nu)}{G(\nu)}. \quad (6)$$

Figure 2: Flow diagram of transmission FTIR. Sample and background interferograms are taken. This data is FFTed to produce a spectrogram. A transmittance diagram of the sample is generated by dividing the sample spectrogram by the background spectrogram.

Figure 2 shows the steps necessary to produce a transmittance plot for a sample. The transmittance plot gets very noisy on the ends of the graph. This is caused by the low signal magnitudes at the ends of the spectrum used to generate the transmittance of the sample. Note that the instrument's light source energy fall off at the ends of its optical spectrum, resulting in low signal to noise ratios at the ends of the spectrograms. This could be improved by using a different light source.

Nyquist Sampling Limitation

Since the measurements of $I(p)$ are sampled at discrete mirror positions, there is a maximum light frequency (wavenumber) that the spectrometer can measure without aliasing. This frequency follows from the Nyquist sampling theorem [6] that when a signal is sampled, the sampling rate has to be twice the maximum frequency present in the signal in order to reconstruct the original signal. The maximum frequency that can be resolved with a typical commercial system results from the mirror position

control system, which uses the interference from an auxiliary HeNe laser. A sample is taken at every zero crossing of the HeNe interference pattern. This means that the detector takes a sample every time the mirror travels 316 nm. If the Nyquist sampling theorem is applied for this sampling rate, a maximum wavenumber of 15798 cm^{-1} can be resolved with a typical FTIR spectrometer.

FTIR Modes of Operation

Generally, there are two ways of controlling the optical path length difference of the interferometer, continuous scan mode and step-scan mode. Continuous scan mode is often used for steady-state measurements and step-scan for dynamic TRS measurements.

Continuous Scan Mode. One method of controlling the mirror is by operation in a continuous scan mode, where the velocity of the moving mirror is held constant. The sinusoidal interference pattern generated by a reference HeNe laser is used to control the optical path length difference of the moving mirror. The mirror's position can be determined by evaluating (2) for the narrow band HeNe reference laser. The photodetector's amplified signal of the HeNe reference's interferogram is AC coupled to a zero crossing detector circuit, which causes a sample to be taken at each zero crossing as the mirror moves at a constant rate. This type of control is very precise and very repeatable due to the stability of the HeNe frequency. Moreover, this method is not restricted to the HeNe zero crossings because an electronic phase lock loop (PLL) can be used to increase the number of zero crossings beyond the HeNe reference signal to resolve higher optical frequencies.

Step-Scan Mode. The other method of controlling the path length difference is step-scan mode. Here, the mirror is controlled to a sequence of fixed positions, or fixed

steps of p , allowing intensity information to be collected over time at each position. This kind of control is needed for TRS experiments, where an impulse of light is sent out and the transient of the sample is measured for each discrete p . This type of control is much harder to do, because the HeNe interference signal detected while the mirror is stopped will be a constant DC level and therefore cannot be AC coupled to determine the zero crossings. The DC offset value would have to be estimated and subtracted from the HeNe interference signal to determine the zero crossings. The mirror is then stopped at the estimated zero crossings of the HeNe feedback signal.

If the laser intensity drifts during a step-scan experiment, there will be error in the stopping mirror positions creating an error in the measurement. This method is limited to sampling at the zero crossings of the reference HeNe signal. The PLL method of increasing the sampling rate will not work for this mode of operation, since a PLL is designed to work over a small range of frequencies. When the mirror stops in step-scan mode, the PLL is not able to lock onto the DC signal and it fails to work.

Time Resolved Spectroscopy Experiments

There are several methods for doing time resolved spectroscopy (TRS) [8]. Some methods use a continuous scanning approach because it is easier and more repeatable to control the mirrors velocity and take samples at regular spaced intervals as compared to step-scan mode. However, there are many limitations involved in the continuous scan approach, making the step-scan method the preferred solution.

Rapid and Ultrarapid TRS Measurements

Rapid and ultrarapid scanning is a technique where spectrograms are generated very quickly, one after the other and displayed with respect to time. The idea behind

the rapid and ultrarapid scanning is that the spectrograms are captured one after another using a continuous scanning approach. This is done very quickly as compared to the transient event, so it is as if the spectrograms are taken instantaneously over a period of time. When the spectrograms are aligned in the order in which they were taken, TRS properties can be seen. For this method, a pulse of light is generated and fast spectrograms are generated. The mirror velocity is much greater than in a typical IR spectroscopy experiment so the scanning time is shorter, allowing for more spectrograms to be taken in a given amount of time. The scans are repeated until the transient event is over. This method is good for slow transient events, due to the minimum time required to complete a full mirror scan. Fast dynamic events cannot be resolved due to the constraint that the scanning time has to be at least an order of magnitude faster than the time sampling rate. An ordinary, analog PLL could be used for this class of experiments to increase the optical frequency range.

Stroboscopic TRS Measurements

Stroboscopic sampling is a technique for collecting time and frequency data at the same time. The mirror is scanned at a constant velocity, as time samples are being generated at evenly spaced samples of the interferogram. During each scan, a pulse of light is generated. The timing of the light pulse is advanced by one sample time, relative to the start of the scan, for each successive scan, so a full time resolved interferogram can be generated by arranging the sample order. This technique can measure faster dynamic events than the rapid and ultra rapid scans but the decoding of the data is complicated and the measurements take much longer than the rapid or ultrarapid scans.

Step-Scan TRS Measurements

The step-scan method is good because it decouples the time and frequency data of a TRS. This means that time resolution is independent of the frequency resolution. For step-scan mode, the mirror is stopped at even spaces while a pulse of light is generated and transient information is taken. This is done for a series of mirror positions, giving both time and frequency data. The draw back to this method is it is very hard to stop the mirror in one place without it moving.

Overview of Proposed Position Control

Some fast kinetic experiments require wavelengths of around 200 nm to be resolved. Continuous scanning methods of TRS can achieve this spectral range but the time resolution is not fast enough to record the dynamic event. The step-scan method can record the time data fast enough but it is limited to the HeNe sampling intervals. Assuming that the detector is useful at higher frequencies, the analytical range of a step-scan experiment can be extended by controlling the mirror to values of p that are more finely spaced than the HeNe interference. Theoretically, this could be achieved by using a higher frequency reference laser than the conventional HeNe, but cost and compatibility with existing optics make this solution impractical. One approach [5] that has been used is to stop the mirror at the minima and maxima of the reference laser interference, in addition to using zero crossings, to effectively double the FTIR frequency range. To do this, the mirror is dithered slightly to provide an excitation signal so custom electronics can lock to the maximum and minimum of the HeNe reference signal. Although this technique is useful for signal-averaging step-scan applications, the slight dithering motion would introduce an error in a TRS experiment.

This thesis proposes a modification of existing FTIR path-difference control systems to extend the frequency range of the instrument in step-scan mode, without dither. In the following chapter, the FTIR is modeled as a position control system. The feedback in the model consists of the HeNe reference laser interferogram signals, from which we propose to estimate the path length difference. The FTIR control is modified by synthesizing feedback signals that cause the instrument to step on a finer grid. Chapter 3 describes the implementation of using a digital signal processor (DSP) to estimate the mirror's position and the interfacing electronics to connect the DSP module in the feedback loop. Chapter 4 shows the performance of system as installed in a Brüker FTIR. Finally, Chapter 5 discusses the results and gives some ideas to further improve a step-scan experiment.

APPROACH AND METHOD

A schematic diagram of the HeNe mirror position control system of the Brüker IFS 66 FTIR spectrometer is shown in Fig. 3. It differs from Fig. 1 in that there is an additional path and detector for the HeNe mirror position control system. The moving mirror is actuated by a voice coil, which is driven by the Brüker control system. The control system uses the feedback from detectors y_1 and y_2 to estimate the mirror position. The theoretical detector output is derived from (2), where $G(\nu)$ is an impulse at ν , with an amplitude A for the HeNe laser. Integrating (2) for the HeNe gives,

$$I_{HeNe}(p) = \int_0^\infty G(\nu_{HeNe})(1 + \cos(2\pi\nu p))d\nu = A(1 + \cos(2\pi\nu_{HeNe}p)). \quad (7)$$

Taking into account the extra path length difference for y_2 , the output of y_1 and y_2 is

$$\begin{aligned} y_1 &= A(1 + \cos(2\pi\nu_{HeNe}p)) \\ y_2 &= A(1 + \cos(2\pi\nu_{HeNe}(p + \Delta))), \end{aligned} \quad (8)$$

where p is the path length difference for detector y_1 , Δ is the extra path length difference for detector y_2 , and ν_{HeNe} is the HeNe wavenumber. The interferometer is calibrated optically so that the detectors are roughly in quadrature, i.e.

$$\begin{aligned} y_1 &= A(1 + \cos(\phi)) \\ y_2 &= A(1 + \cos(\phi + \phi_0)), \end{aligned} \quad (9)$$

where ϕ is the effective angle corresponding to mirror position and ϕ_0 is the constant phase difference between detectors y_1 and y_2 , which is calibrated to be close to ninety degrees.

Figure 3: Schematic diagram of mirror position control system in the Brüker FTIR spectrometer.

Feedback Dynamics

Figure 4 shows the FTIR position control in Fig. 3 modeled as a classical control system. The section to the left of the dashed line represents the part in Fig. 3 labeled “Brüker Control System”. The input $p_c(t)$ in Fig. 4 is a mirror position control command corresponding to the PC input of Fig. 3. The Brüker control system estimates the mirror’s position, which is represented by the box labeled P.E., and generates an error signal $e(t)$ by subtracting the mirror position estimate from the PC position command. The error signal is fed into an amplifier, with a gain $K(s)$, which drives the voice coil. The voice coil can be modeled with a single time constant [2].

Analysis of the FTIR feedback system of Fig. 4 can be considerably simplified with a few reasonable assumptions. First, if the HeNe mirror position control system in the FTIR is assumed to work properly, then the feedback path from $p(t)$ to $\hat{p}(t)$

Figure 4: FTIR mirror position control as a feedback system.

has a gain of unity, i.e.

$$\hat{p}(t) = p(t). \quad (10)$$

Continuing with this assumption, the open loop gain, in the Laplace domain is

$$L(s) = \frac{\hat{P}(s)}{E(s)} = K(s) \frac{\alpha}{s(\tau s + 1)}. \quad (11)$$

The closed loop transfer function from $p_c(t)$ to $p(t)$ is completely determined by the loop gain $L(s)$. From the diagram,

$$\begin{aligned} P(s) &= L(s)E(s) \\ &= L(s)[P_c(s) - P(s)]. \end{aligned}$$

The closed loop transfer function is therefore

$$\frac{P_c(s)}{P(s)} = \frac{L(s)}{1 + L(s)}. \quad (12)$$

Substituting (11) into (12), the closed-loop transfer function can be rewritten as

$$\frac{P_c(s)}{P(s)} = \frac{K(s)\alpha}{\tau s^2 + s + K(s)}. \quad (13)$$

The closed loop transfer function is significant because it determines how the mirror position $p(t)$ responds to a command input $p_c(t)$. The roots of the denominator of

this transfer function are the natural frequencies of the closed loop system. If the natural frequencies are complex conjugate symmetric, the real part of the natural frequencies is $\frac{-1}{2\tau}$. The step response of the system, based on this model, should have a damped sinusoidal part decaying with an envelope of $e^{-t/2\tau}$.

Simulations with constant $K(s)$ and $\tau = .9$ ms show fairly close agreement with measurements taken from our instrument. The upper graph in Fig. 5 shows an oscilloscope image of the detector outputs during a step response, while the lower graph shows simulated detector outputs with a transfer function based on (13). There are very likely unmodeled dynamics in the actual system, e.g. the electrical dynamics of the voice coil, but the mechanical time constant τ dominates the response. The closed loop model provides a good understanding for the dynamics of the mirror control system.

Approach For Enhanced Position Control

The proposed technique for increasing the mirror position resolution of the spectrometer is shown in control diagram form in Fig. 6. The idea is to intercept the easily accessible HeNe interferometer detector outputs, compute an estimate $\hat{\phi}(t)$ of the angle $\phi(t)$, and substitute for the intercepted signals with quadrature “synthetic” detector signals with an new angle $N\hat{\phi}$ where N is a small integer.

The analysis of the modified control system is straightforward. The gain from the mirror position $p(t)$ to $\hat{p}(t)$ is N rather than unity, i.e.

$$\hat{p}(t) = Np(t). \quad (14)$$

Similarly, the loop gain $L(s)$ increases by a factor of N ,

$$L(s) = \frac{NK(s)\alpha}{s(\tau s + 1)}, \quad (15)$$

Figure 5: Measured and simulated detector output waveforms. Simulated detector outputs are shifted to correspond with the oscilloscope display

Figure 6: FTIR mirror position control modified for finer spatial resolution.

and the closed loop transfer function becomes

$$\frac{P(s)}{P_c(s)} = \frac{1}{N} \left[\frac{L(s)}{1 + L(s)} \right]. \quad (16)$$

In terms of the FTIR behavior, the ideal closed loop gain in Fig. 6 is $\frac{1}{N}$, so when $N = 1$ the mirror position p ideally steps as commanded by $p_c(t)$. If $N = 2$, then the steps in the actual mirror position p are half that of the command, providing finer spatial sampling and enhanced frequency range. Inserting a gain in the feedback path appears to fix the resolution problem, but it reduces the “degree of stability” of the closed loop system. The stability will be discussed in Chapter 3.

Modeling and Estimation

The idealized detector outputs in (9) differ from true detector outputs on the FTIR by electronic offsets and gain imbalances between the two sensors. A more realistic model of the electronic detector outputs is

$$\begin{aligned} y_1 &= A \cos(\phi) + b_1 \\ y_2 &= A\delta \cos(\phi + \phi_0) + b_2, \end{aligned} \quad (17)$$

where b_1 and b_2 lump the DC term of the ideal response with the electronic offset for each detector, and δ is a gain-mismatch between the detectors. Figure 7 shows

a graphical representation of the parameters in equations (17). y_1 is plotted with respect to y_2 for a complete cycle of the moving mirror. The elliptical shape is due to the gain-mismatch in the detectors, the rotation of the ellipse is due to ϕ_0 , and the center of the ellipse represents the DC offset values. All of the parameters from (17) are time varying. A , δ , b_1 , b_2 and ϕ_0 can be thought of as varying slowly with time. The angle ϕ changes on a much faster time scale due to the movement of the mirror. For an estimator to find all of these values at once, a constraint on how these parameters change with respect to one another has to be made. One way to estimate the parameters of (17) is to set up two coordinate systems and estimate the center of the ellipse on a cartesian coordinate system and the angle ϕ on a polar coordinate system. This allows the estimates of each coordinate system to evolve at two different rates. The time constants of the estimator need to be chosen so that the parameters do not drift around when the mirror is stopped. This type of estimation involves two time constants, one for ϕ , and one time constant for the slow varying parameters, A , δ , b_1 , b_2 and ϕ_0 .

This two-time constant approach can be avoided by realizing that ϕ does not have to be estimated. The smoothing effect the estimation has on ϕ is not required because the control system is able to tolerate and reject some noise. Indeed, the phase lag of the estimate would tend to destabilize the control loop. This allows for ϕ to be calculated directly from the measurements y_1 and y_2 and reduces the order of the estimator.

The ellipse can be projected onto a circle by mapping y_1 and y_2 to a different coordinate system. By recognizing that the cosine term in y_2 can be written as $\cos(\phi)$ and $\sin(\phi)$, (17) becomes

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = M \begin{pmatrix} A \cos(\phi) \\ A \sin(\phi) \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad (18)$$

Figure 7: This represents the mirror parameterization from (17). The idea is to estimate b_1 , b_2 , r , and ϕ at the same time.

Figure 8: This shows how the detector signals y_1 and y_2 are mapped to q_1 and q_2 .

where M contains δ and ϕ_0 information to rotate and stretch an ideal circle to the rotated elliptical shape of Fig. 7 and b_1 and b_2 shift the center of the ideal circle to the first quadrant, as in Fig. 7. This model can be recast as

$$q = A \begin{pmatrix} \cos(\phi) \\ \sin(\phi) \end{pmatrix} = M^{-1} \left(\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right) = \begin{pmatrix} 1 & 0 \\ a & b \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} - \begin{pmatrix} c \\ d \end{pmatrix}. \quad (19)$$

This equation represents the projection of the measurements into an ideal coordinate system where the path difference is mapped to a point q on a circle with radius A , centered on the origin. Figure 8 shows the mapping graphically between the true detector outputs and the projected outputs.

To reduce the order and allow for a slow varying, single time constant estimator, an estimate that does not depend on ϕ can be created. Since $q^T q = A^2$,

$$\begin{aligned} A^2 = & (1 + a^2)y_1^2 + b^2y_2^2 + 2aby_1y_2 + 2(c + da)y_1 + \\ & 2db y_2 + c^2 + d^2. \end{aligned} \quad (20)$$

This equation can be written as a linear identification problem with

$$\begin{pmatrix} y_1^2 & y_1 y_2 & y_1 & y_2 & 1 \end{pmatrix} \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \end{pmatrix} = -y_2^2, \quad (21)$$

where

$$\begin{aligned} \mu_1 &= \frac{1 + a^2}{b^2} \\ \mu_2 &= \frac{2a}{b} \\ \mu_3 &= \frac{2(c + da)}{b^2} \\ \mu_4 &= \frac{2d}{b} \\ \mu_5 &= \frac{c^2 + d^2 - A^2}{b^2}. \end{aligned} \quad (22)$$

Given sufficient excitation, the parameters μ_k can be determined by standard methods, for example, recursive least squares. Similarly, the coefficients of the mapping equation (19) can be determined algebraically from the μ_k . If all the quantities to be estimated vary slowly compared to the mirror position, then the parameter update can be done on a convenient time scale. It is only necessary to evaluate (19) in real time to find the latest q from y_1 and y_2 for purposes of control.

Given q , normalized synthetic detector outputs u for an integer angle multiplication factor N are given by the real and complex parts of the rotation

$$u_1 + ju_2 = \frac{(q_1 + jq_2)^N}{|q|^N}. \quad (23)$$

In general, practical values of N are two or three. Detector and optic limitations generally make further extension of the frequency range impractical.

Figure 9: A software flow diagram for the DSP.

A flow diagram for the “New Module” in Fig. 6 is shown in Fig. 9, where the Estimator box corresponds to (21) and (22), the projection box corresponds to (19) and the q^N box corresponds to (23).

Properties of The Estimate

The proposed scheme computes an estimate $\hat{\phi}$ of the effective angle ϕ of the interferometer path length difference and outputs signals with $N\hat{\phi}$ to the rest of the control system. While the mirror control system may be able to reject some zero-mean noise in this feedback path, any offset value or bias in $\hat{\phi}$ would cause an error in the mirror position. Therefore, it is important to ensure that the angle represented by q in (19) is unbiased with respect to sensor noise in the inputs y_1 and y_2 . A computational technique known as the bootstrap [10, 11] can be used to handle the complexity of how sensor noise would influence the computation of q through (21), (22), and (19). A bootstrap dataset consists of data plus synthetic sensor noise with some assumed statistical properties. Applying the estimator to many datasets with different noise samples builds an approximation to the distribution of quantities of interest, i.e. $\hat{\phi}$.

Figure 10a shows two bootstrapped distributions for the error in the angle $\hat{\phi}$. The outer distribution shows $\hat{\phi} - \phi$, where $\hat{\phi}$ is the angle of q in (19) and ϕ is the true angle. The value of q was determined by estimating μ with (21) and solving for the coefficients a, b, c, d with (22). The inner distribution represents the angular error when the estimation step is eliminated, i.e. using perfect values of a through d in (19).

Figure 10: Histogram of 20,000 bootstrap estimates for the residual $(\phi - \hat{\phi})$ with $SNR = 67$. The mean value for the outer distribution is $-5.3e - 04$ and the mean for the inner distribution is $-3.1e - 05$.

The distributions in Fig. 10a are generated from one angular position after there has been sufficient excitation for μ_k , but are representative of distributions at other angles. The true model parameters and noise used for the bootstrap are representative of the actual Brüker parameters and noise. The SNR for the bootstrap data sets is 67. The bootstrap distribution for each angle is proven to be zero-mean confirming that the parameterization and estimate from (21) is unbiased and produces good estimates of $\hat{\mu}$.

Figure 10b shows the convergence of the log of the mean squared error $\hat{\phi} - \phi$ as a function of mirror position for step-scan mode with $N = 3$. After a few steps, the proposed method has sufficient excitation to produce q values with angles that have a variance close to that of the additive noise, as shown in Fig. 10b as a horizontal line at -4.

Estimating the projection parameters and using them to project the measurements, y_1 and y_2 , onto the ideal coordinate system has given the ability to calculate ϕ instead of estimating it. This significantly reduced the complexity of the calculation and greatly simplifies the choice of the time constant. The mapping approach appears to provide an unbiased value of $\hat{\phi}$.

IMPLEMENTATION

A DSP module, consisting of interface circuitry and a DSP, was made to carry out the technique proposed in Chapter 2. The DSP is used to calculate $\hat{\phi}$ and the synthetic detector output signals with an angle of $N\hat{\phi}$. A picture of the DSP module is shown in Fig. 11. Slight adjustments of the Brüker control system were needed to run the DSP module in the control loop.

Hardware

The new module in Fig. 6 consists of two printed circuit boards (PCBs) stacked on top of one another. The bottom board is a LF2407A eZdsp evaluation board from Texas Instruments. The DSP is a TMS320LF2407A 16 bit, 40 MIPS, control oriented DSP with a built in 16 channel, 10 bit analog to digital converter (ADC) and a serial communication interface to easily send data to a digital to analog converter (DAC). The ADC is capable of sampling at 500 kHz, however, 80 kHz is the sample rate that is used. The top PCB is the analog interface board, which has the input circuitry, output circuitry, DAC, and the DSP loop switch. A schematic of the analog interface board is shown in Fig. 12. The input circuitry amplifies and conditions the signals from detectors y_1 and y_2 for the ADC. The circuitry consists of a cascade of operational amplifiers (op amps) to convert the current from detectors y_1 and y_2 , to a voltage. The first stage is a traditional current to voltage inverting op amp. The second stage is a standard inverting op amp which is used to lower the cut off frequency and provide additional gain if needed. The output circuitry uses a Howland current source to convert the signals from the DAC into signals which look like they are coming from detectors y_1 and y_2 . The section labeled “Output Circuitry” in Fig. 12 shows the

Figure 11: A picture of the DSP module used to estimate the mirror position and move the mirror on a finer grid.

schematic diagram for the photodiode “look-alike” circuit. For a typical Howland current source, the resistors are chosen so that the ratio of the resistors connected to the inverting node is equal to the ratio of the resistors connected to the non-inverting node. If this condition is met, the output current is the difference between the input voltage from the switch and the ground node voltage, divided by the input resistor connected to the non-inverting node and the switch, providing a differential input and an output current that does not depend on the load resistance. The typical problem with a Howland current source is that the resistors are not perfectly matched. This means that the input voltage to the Howland current source no longer has the differential advantage and the load resistance can change the gain of the current source. However, this is not a problem in this application due to the fact the current is being fed into the Brüker electronics with a very low input impedance eliminating

Figure 12: A schematic of the interface circuitry for the DSP module.

the load impedance problem. Moreover, the gain does not have to depend strictly on the input resistor since the gain and offsets are being estimated by the Brüker control system. The DAC is a Texas Instrument TLV5638 with 12 bits of resolution, dual output, and a serial input. The DAC is capable of output sample rate of 625 kHz, however, 80 kHz is the output sample rate that is used. The DSP loop switch, in Fig. 12, is used to switch the DSP module between two modes. “Pass through” mode allows the detector signals y_1 and y_2 to be passed through as if the DSP was not in the loop. This allows the DSP module to be in the loop without affecting the current operation of the Brüker control system. The other mode of operation is $N=3$ mode, where the DSP is in the loop and the output of the module is the synthetic detector signals u_1 and u_2 .

Software

To calculate the synthetic detector signals u_1 and u_2 , the DSP first estimates μ from (21), using RLS with a forgetting factor. A derivation of RLS can be found in [3] and [9]. The basic form of the state update given the measurement model $z_k = \varphi'_k \mu_k$ is

$$\hat{\mu}_{k+1} = \hat{\mu}_k + P_k \varphi_k (z_k - \varphi'_k \hat{\mu}_k), \quad (24)$$

and the inverse covariance update is

$$P_k^{-1} = \lambda P_{k-1}^{-1} + \varphi_k \varphi_k', \quad (25)$$

where λ is the forgetting factor and ranges from 0 to 1. A forgetting factor is commonly used to ensure that the state continues to track the input. Without a forgetting factor, the inverse covariance integrates to a very large number and the covariance goes to zero. The state update stops adding any new information from the measurement z_k and the update stops evolving. For a time-varying process, the forgetting factor λ allows the newest measurements to affect the update.

The detector outputs from y_1 and y_2 were measured over a period of time to find how the parameters of (17) varied with time. To do this experiment, the interferometer was set to continuous scan mode so the output of detectors y_1 and y_2 were 2.2 kHz sine waves. A National instruments PXI-5122 14-bit DAC card, sampling at 200 kHz, was used to measure the outputs from detectors y_1 and y_2 . The room temperature was also measured to see if it had any effect on the parameters. A two-second interval of temperature and detector output data was collected every five minutes, with a total test run time of 13 hours. During each two-second interval of data collection, the temperature for that interval was estimated using least squares with a constant temperature model. The parameters for the detector outputs were estimated using a nonlinear Gauss-Newton parameter estimation for all the parameters of (17). The model used for the estimate assumes that the mirror is moving at a constant speed. Each two-second interval of detector output data was reduced to the 20,000 samples where the mirror was moving at its most constant velocity (note that when the mirror changes directions, the velocity changes at a high rate and this data should not be used). The Gauss-Newton routine was run for each of the data intervals. Figure 13 shows how the amplitude A of (17) and temperature changes over a period of time.

Figure 13: Amplitude A and temperature change as a function of time.

The other parameters estimated from (17) are not shown, but vary on the same time scale and have a maximum change of 5%. The ambient temperature change, shown in the lower plot, has very little effect on the change in parameters. The two plots have very little correlation. However, when the temperature reaches a steady value, the amplitude, A , is also fairly constant. The key thing to note is the amplitude has a time constant of approximately 45 minutes and changes a maximum of 5%. This is used to determine the forgetting factor for the parameter estimate.

When deciding the correct forgetting factor, there is a tradeoff between how long the mirror can sit in one place without excitation and being able to adapt to the changing parameters. Lack of excitation requires a slow forgetting factor. If the system parameters change rapidly, a fast forgetting factor is needed. Based on Fig. 13 and the experimental requirements, a forgetting factor of 15 minutes is used.

Software Implementation

To implement RLS with the forgetting factor in the DSP, the steady-state solution of the covariance was found so the inverse covariance does not have to be computed at each update. This allows the DSP to only update (24) and makes it possible to compute the estimate of $\hat{\mu}$ in real time. The steady-state value was found by simulating the detector signals y_1 and y_2 in Matlab and estimating μ for the simulated detector signals until the covariance reached a steady-state value. The steady-state value of the covariance was taken with the forgetting factor of 15 minutes used in this simulation.

After the DSP has estimated μ , the DSP finds the mapping parameters from (19). The mapping parameters could be solved algebraically, however, due to the speed and quantization error from calculating the parameters, an iterative scheme is used to find the mapping parameters. The iteration is derived from a first order Newton's method for finding the roots of an equation. The Newton update is derived from the Taylor series expansion of a function for perturbations v about a point x , i.e.

$$f(x + v) = f(x) + \nabla f(x)^T v. \quad (26)$$

Setting $f(x + v) = 0$ and solving for v gives

$$v = -(\nabla f(x)^T)^{-1} f(x) = -J^{-1} f(x), \quad (27)$$

where J is the Jacobian of $f(x)$ and the step v minimizes the first order Taylor series approximation of (26). This can be written as an update,

$$x_{k+1} = x_k - J^{-1} f(x), \quad (28)$$

where the roots of $f(x)$ are being estimated. This iteration scheme is used for inverting

and finding the parameters of (22). The equations to solve are

$$\begin{aligned}
\mu_1 &= \frac{1 + a^2}{b^2} \\
\mu_2 &= \frac{2a}{b} \\
\mu_3 &= \frac{2(c + da)}{b^2} \\
\mu_4 &= \frac{2d}{b} \\
\mu_5 &= \frac{c^2 + d^2 - A^2}{b^2}.
\end{aligned} \tag{29}$$

The system can be multiplied by b^2 so that the divide does not have to be carried out. This is done because the DSP does not have a hardware divider and the software divide takes many cycles. The system of equations becomes

$$f(a, b, c, d) = \begin{pmatrix} \mu_1 b^2 - (1 + a^2) \\ \mu_2 b^2 - 2ab \\ \mu_3 b^2 - 2(c + da) \\ \mu_4 b^2 - 2db \end{pmatrix}. \tag{30}$$

The Jacobian of $f(a, b, c, d)$ is

$$J = \begin{pmatrix} -2a & 2\mu_1 b & 0 & 0 \\ -2b & 2\mu_2 b - 2b & 0 & 0 \\ 2d & 2\mu_3 b & 2 & 2a \\ 0 & 2\mu_4 b + 2d & 0 & 2 \end{pmatrix}. \tag{31}$$

Substituting (30) and (31) into (28) gives

$$\begin{pmatrix} \hat{a}_{k+1} \\ \hat{b}_{k+1} \\ \hat{c}_{k+1} \\ \hat{d}_{k+1} \end{pmatrix} = \begin{pmatrix} \hat{a}_k \\ \hat{b}_k \\ \hat{c}_k \\ \hat{d}_k \end{pmatrix} - J^{-1} f(\hat{a}_k, \dots, \hat{d}_k). \tag{32}$$

To avoid having to take the inverse of the Jacobian matrix for each update, the

Jacobian can be approximated with a fixed matrix, i.e.

$$J = \begin{pmatrix} 0 & 2 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}. \quad (33)$$

Using this suboptimal gain in (32), the DSP can converge to the correct value within a few samples. The new estimate becomes

$$\begin{pmatrix} \hat{a}_{k+1} \\ \hat{b}_{k+1} \\ \hat{c}_{k+1} \\ \hat{d}_{k+1} \end{pmatrix} = \begin{pmatrix} \hat{a}_k \\ \hat{b}_k \\ \hat{c}_k \\ \hat{d}_k \end{pmatrix} - \begin{pmatrix} 0 & -.5 & 0 & 0 \\ .5 & 0 & 0 & 0 \\ 0 & 0 & .5 & 0 \\ 0 & 0 & 0 & .5 \end{pmatrix} f(\hat{a}_k, \dots, \hat{d}_k). \quad (34)$$

Using this result, q is calculated with (23) to give the projection for y_1 and y_2 to the ideal coordinate system. The synthetic detector signals u_1 and u_2 are calculated with (23). The calculated DC offsets are added back to u_1 and u_2 and sent to the DAC for output.

Operational Procedure and Experimental Setup

The interferometry experiments were done using a Brüker ISF 66/S interferometer with the step-scan option. The interferometer is controlled by a PC using Opus 3.0.1 software running on OS/2. The setup begins by inserting the DSP module in between detectors y_1 and y_2 and the Brüker control system and setting the DSP module to “pass through” mode. With the DSP module in the loop, the Brüker mirror is set to a slow, continuous dither so the output from detectors y_1 and y_2 is a 1.6 kHz sine wave. This allows the DSP to adapt its coefficients to the correct values. Once the DSP has estimated the correct parameters, the moving mirror is positioned for the start of the step-scan experiment and waits for the start command from the PC. The

DSP module is then switched to N=3 mode and a step-scan experiment is started with the PC interface software.

The Brüker control board has a proportional integral derivative (PID) compensator to change the mirror dynamics in step-scan mode. The gain of each element of the PID compensator is easily adjusted with trim pots located on the Brüker control PCB. With the DSP in N=3 mode the loop gain of Brüker control system is increased by a factor of three as mentioned earlier in Chapter 2. This increase in loop gain reduces the stability of the closed loop system by reducing the phase margin of the system. Another source of instability is the delay that the DSP module adds to the system. The delay also decreases the phase margin of the control system. These effects and other higher-order effects tend to make the control system unstable. To correct for instabilities of the system, the gain of the loop can be reduced by a third by reducing all the elements in the PID controller along with slight tweaking of each element to provide a slight increase in the phase margin.

To adjust the elements of the PID controller, an oscilloscope is used to monitor the output of detectors y_1 and y_2 . To adjust the mirror dynamics, the DSP module is put into N=3 mode and a step-scan experiment is started. If there is any oscillation of the mirror, when the mirror is supposed to be at a fixed position, or if the overshoot and settling time are not good enough, the PID gains of each element are adjusted until the desired step response is achieved. After the dynamics have been adjusted to give a good step response, the mirror can be moved back to the starting position and more step-scan experiments can be run.

How long the mirror sits in one place without any excitation is a potential concern. If the mirror sits at the starting position or any other position long enough, the estimate of the mapping equation in (19) will drift, which will produce erroneous values of u_1 and u_2 . The period of time spent at each position is short enough that

the effects of this drift are minimal for a normal step-scan experiment.

The open loop and closed loop results of the DSP module in the control loop will be discussed in the next chapter, along with experimental data for a step-scan experiment.

EXPERIMENTAL PROCEDURE AND RESULTS

To test the performance of the DSP module in the loop, multiple experiments have been performed using the FTIR spectrometer. All of the experiments were done using the same sample test specimen so the data can be easily compared. The testing consisted of a set of measurements for the sample with the DSP module in “pass through” mode and a set of measurements with the DSP module in N=3 mode. The experiments done in pass through mode are used to show the current performance and measurement capabilities of the interferometer. The other experiments show the performance and advantages of using the N=3 mode of the DSP module.

Closed Loop Mirror Positions

The input signals, y_1 and y_2 , to the DSP module were measured with a 14-bit National Instruments PXI-5122 DAC card for a step-scan N=3 experiment to test the stopping positions of the closed loop control system. The sample rate of the DAC card was 100k Hz and data was taken for 15 seconds, allowing the mirror to step through 81 mirror positions. The steady state values of each mirror position were taken and used in a least squares estimate to find the gain and offset parameters for the entire data set. To estimate the parameters, it was assumed that the mirror stopped at evenly spaced positions for each mirror step.

The stopping mirror positions for the closed loop system are shown in Fig. 14. The clusters of dots show where the mirror is stopping. The dashed line represents the ideal mirror position as it moves. The cross marks on the graph are generated from the least squares fit and represent ideal, evenly spaced, mirror stopping positions. This figures shows that there is a bias in the fixed mirror positions. It is unclear, in

Figure 14: This figure shows the ideal stopping mirror positions along with the actual stopping positions. The cross arrows show the ideal mirror positions and the clusters are where the mirror actually stopped. The dotted line is the ideal mirror path.

this closed loop experiment, whether the bias comes from the DSP method or other parts of the loop.

DSP Mirror Estimate

The DSP estimation routine was checked by measuring the input and output signals of the DSP module, y_1 and u_1 respectively, and checking to see if the output signal u_1 represented three times the angle of y_1 . Figure 15 shows a block diagram of the test that was done to check if the DSP was introducing angular error. Signals y_1 and u_1 were measured with the National Instruments PXI-5122 DAC card sampling at 100k HZ for a step-scan $N=3$ experiment. A Matlab model of an ideal DSP was

Figure 15: A block diagram for a test that was done to check if the DSP module was producing biased mirror positions. The DSP module was tested against an ideal DSP model.

created generating an output signal $\hat{u}_1$ with three times the angle of the input signal y_1 . Figure 16 shows the results of the computed value $\hat{u}_1$ as compared to the actual DSP output u_1 . The figure shows that the output of the DSP module is three times the frequency of the input signal. The error distribution in Fig. 16 shows the DSP module output minus the modeled DSP output. This value is zero mean, showing that the DSP module is estimating the correct value.

Spectral Experiment

The Brüker interferometer uses a tungsten broad band light source for the interferometer measurements. To do the N=3 step-scan experiment, blue optical filters are used to reduce the intensity of the red and infrared light so a larger aperture could be used without saturating the interferometer's photodetector. The larger aperture and blue optical filters allow more of the higher frequency light into the interferometer, increasing the signal to noise ratio of the higher frequency light.

To run a step-scan experiment, the DSP module is switched to pass through mode and the interferometer's alignment software is run to align the fixed mirror. The aperture is set to 6 mm to allow the maximum amount of light through the

Figure 16: Measured DSP output and model output in 3x mode. The empirical distribution of errors is very close to zero mean.

interferometer. To do a full experiment, two step-scan experiments are run back to back to obtain a spectrum of a sample and a background measurement without a sample. The two measurements are used to generate a transmittance spectrum for the sample of interest. For this experiment, an erbium doped crystal was used for the sample. After the mirror is aligned and the DSP module has adapted to the parameters, the mirror is stopped, the DSP is switched to N=3 mode, the erbium crystal is placed in the sample chamber, and the step-scan experiment is started. Once this experiment has finished, the erbium crystal is removed from the sample chamber and another experiment is done for the background sample.

A static N=3 step-scan experiment was done on an erbium crystal to test the performance of the DSP module. The static experiment was done because it can be compared to a continuous scan experiment, which uses the electronic PLL to measure higher frequencies. The extended spectral features in a TRS experiment, in N=3 mode, can not be compared to anything else since a step-scan experiment done in pass through mode can only measure frequencies up to 15798 cm^{-1} . Figure 17 shows the spectrogram and transmittance properties for a step-scan N=3 experiment with an erbium crystal. The dashed line marks the unmodified step-scan maximum frequency. The features to the left of the dashed line cannot be resolved with the traditional step-scan techniques. These features would be aliased to the lower frequencies if this experiment was repeated with the DSP in "pass through" mode. A spectrogram and transmittance diagram for a continuous scan is shown for the same erbium crystal, which is used as an ideal spectrogram and transmittance diagram to compare against the step-scan N=3 experiment. The transmittance properties of the step-scan experiment match that of the continuous scan experiment very closely. There is some deviation at the higher frequencies. Note that the photodetector's response falls off at the upper end of the spectrogram for the continuous scan experiment. This is due

to the scanning velocity of the experiment and the response of the photodetector. Furthermore, the ends of the transmittance spectrum are noisy due to the low optical power, resulting in a much lower signal to noise ratio in that region.

Figure 17: The lower section of this graph contains four plot lines showing the background spectrograms (the two upper lines) and the erbium crystal sample spectrograms (two lower lines). In the upper section of this graph, the transmittance spectrum of the erbium crystal sample is shown as determined in step and continuous scan modes. The step-scan data was collected in N=3 mode and the continuous scan data was collected utilizing the PLL with the DSP in “pass through” mode.

CONCLUSION

The DSP module extends the frequency range for a standard FTIR spectrometer with little modification to the existing instrument. The DSP module plugs into the feedback control system, computes the mirror position using the existing HeNe feedback control signal, and supplies synthetic mirror position signals to the existing Brüker control system. The DSP module produces very good results in N=3 mode for step-scan experiments. The broadband light source used for the experiment was not ideal for testing the full frequency range that the DSP module allows for. However, the results show an improvement in the frequency range compared to a standard step-scan interferometer.

Further improvements could be made with a better understanding of how the Brüker control system calculates the feedback amplitude and offset from the HeNe laser. Measurements indicate that the mirror is stopping at biased positions, while the FTIR is producing essentially perfect outputs. The Brüker control system possibly uses fixed amplitude and DC offset parameter values. With exact knowledge of how the Brüker offsets are calculated, the DSP could correct for the Brüker parameters and force the mirror to stop more consistently. This could improve the quality of the spectra.

The DSP needs sufficient mirror excitation to estimate the correct parameters. The amount of mirror excitation is based on the forgetting factor time constant. By choosing a forgetting factor with a long time constant, the DSP needs less excitation and the mirror can stay in one position for a long time. However, if the estimate cannot adapt to the changes in the physical system, the DSP module will produce a poor estimate of the mirror position. This creates a tradeoff between how fast the

DSP can adapt to the change in the system and how long the mirror can remain in a position without moving.

More testing needs to be done to see how the DSP module performs in a TRS experiment. One way this might be done is to run a TRS step-scan experiment in pass through mode and then again in N=3 mode. Down sampled data from the N=3 experiment should be the same as the pass through experiment.

The results from the first set of experiments are very promising and show that the proposed technique provides finer spatial sampling and increased frequency range for step-scan experiments.

REFERENCES

- [1] ATKINS, P. and DE PAULA, J., *Physical Chemistry*. Freeman, 7 ed., 2002.
- [2] FRANKLIN, G. F., POWELL, J. D., and WORKMAN, M., *Digital Control of Dynamic Systems*. Reading, MA, USA: Addison-Wesley, third ed., 1998.
- [3] GELB, A., *Applied Optimal Estimation*. M.I.T. Press, Cambridge, Massachusetts, 1974.
- [4] JOHNSON, T. and ZACHMANN, G., *Introduction to Step-Scan FTIR*. BRUKER, 2000.
- [5] MANNING, C. J., PALMER, R. A., and CHAO, J. L., “Step-scan fourier-transform infrared spectrometer,” *Rev. Sci. Instrum.*, vol. 62, pp. 1219–1229, May 1991.
- [6] OPPENHEIM, A. V., WILLSKY, A. S., and YOUNG, I. T., *Signals and Systems*. Prentice Hall Signal Processing Series, Englewood Cliffs, New Jersey: Addison Wellesley, 1988.
- [7] SALEH, B. E. and TEICH, M. C., *Fundamentals of Photonics*. New York: John Wiley and Sons, INC., 1991.
- [8] SMITH, G. D. and PALMER, R. A., “Fast time-resolved mid-infrared spectroscopy using an interferometer,” in *Handbook of Vibrational Spectroscopy*, vol. 1, pp. 625–640, 2002.
- [9] STRÖM, K. J. A. and WITTENMARK, B., *Adaptive Control*. Addison-Wesley, second ed., 1995.
- [10] ZOUBIR, A. M. and BOASHASH, B., “The bootstrap and its application in signal processing,” *IEEE Signal Processing Magazine*, vol. 15, pp. 56–76, January 1998.
- [11] ZOUBIR, A. M. and ISKANDER, D. R., *Bootstrap Techniques for Signal Processing*. Cambridge, 2004.

APPENDICES

APPENDIX A

PCB SCHEMATIC

APPENDIX B

DSP CODE


```

*****
* Filename: RLS6_25_05.asm
*
* Author: Brandon Inberg
*
* Last Modified: 06/25/05
*
* Description: Estimates Mirror position for FITR system. Uses RLS to*
*               Estimate rotation matrix and calculates mirror output. *
*               The DSP process is show below.
*****
*
*               \ |      \ | Projection | \ |      \ | U_1,U_2 \ *
* y_1 and y_2 ---| Estimate|---|to ideal sys|---|rotation|----- *
*               | / |-----| / |-----| / |-----| / *
*               |               /|\
*               |-----|
*
*****
;~~~~~
;Global symbol declarations
;~~~~~
        .def start, timer2_isr, adc_isr

;~~~~~
;ref
;~~~~~
        .ref      SINTAB_360

;~~~~~
;Address definitions
;~~~~~
        .include f2407.h

DAC0      .set      0000h      ;EVM DAC register 0 (I/O space)
DAC1      .set      0001h      ;EVM DAC register 1 (I/O space)
DAC2      .set      0002h      ;EVM DAC register 2 (I/O space)
DAC3      .set      0003h      ;EVM DAC register 3 (I/O space)
DACUD     .set      0004h      ;EVM DAC update register (I/O space)
DIPSWCH   .set      0008h      ;EVM DIP switch (I/O space)
LED       .set      000Ch      ;EVM LED bank (I/O space)

;~~~~~
;Constant definitions
;~~~~~
timer2_per      .set      58594      ;250ms timer2 period with a 1/128
                                   ;timer prescaler and 30MHz CPUCLK

pwm_half_per    .set      750        ;period/2, 20KHz symmetric PWM with

```

```

;a 30MHz CPUCLK

pwm_duty      .set      563          ;25% PWM duty cycle

; Constant required for obtaining Taylor Series Approximation of SQRT
__a0          .set      06a48h        ; 0.1037903 scaled by 218
__a1          .set      05d1dh        ; 0.7274475 scaled by 215
__a2          .set      0a9edh        ; -0.672455 scaled by 215
__a3          .set      046d6h        ; 0.553406 scaled by 215
__a4          .set      0bb54h        ; -0.2682495 scaled by 216
__a5          .set      00e5ah        ; 0.0560605 scaled by 216 for Immediate MPY
__isqt2       .set      00b50h        ;(1/sqrt(2))=0.707106 in scaled Q12 format
                                           ; Also sqrt(2) in scaled Q11 format
__isqt2d      .set      05a82h        ;(1/sqrt(2)) in Q15 format
                                           ; Also sqrt(2) in Q14 format

;~~~~~
;Uninitialized global variable definitions
;~~~~~
        .bss      LED_index,1      ;LED index
        .bss      gp1,1
        .bss      gp2,1
        .bss      spi1,1           ;spi general variable
        .bss      spi2,1           ;spi general variable
        .bss      sample1,1
        .bss      sample2,1
        .bss      out1,1
        .bss      out2,1
        .bss      p,15
        .bss      MU,10
        .bss      H,5
        .bss      x,5
        .bss      Y,1
;        .bss      alpha,1
;        .bss      pnorm,1
;        .bss      lam,1
        .bss      E,1
        .bss      bb,1
        .bss      b1,1
        .bss      a1,1
        .bss      c1,1
        .bss      d1,1
        .bss      dhat,1
        .bss      R,1
        .bss      timestwo,2
        .bss      threetimes,2
        .bss      temp,2           ;sin and cos temp variable
        .bss      temp1,2         ;general purpose variable

```

```

        .bsS    out11,1
        .bss    out22,1
        .bss    delta,1
        .bss    X,150
;        .bss    X1,1

dot      .macro  a,b,c,d
        LT      a                ;load treg->a
        MPY     b                ;Preg=a*b
        LTP     c                ;acc=a*b, treg->c
        MPY     d                ;Preg=c*d
        APAC    ;acc=a*b+c*d (Q.15*Q.15=Q.30)
        .endm

fivedot  .macro  a,b,c,d,e,f,g,h,i,j
        LT      a                ;load treg->a
        MPY     b                ;Preg=a*b
        LTP     c                ;acc=a*b, treg->c
        MPY     d                ;Preg=c*d
        LTA     e                ;acc=a*b+c*d, treg->e
        MPY     f                ;Preg=e*f
        LTA     g                ;acc=a*b+c*d+e*f, treg->g
        MPY     h                ;Preg=g*h
;        LTA     i                ;acc=a*b+c*d+e*f+g*h, treg->i
;        MPY     j                ;Preg=i*j
;        APAC    ;acc=a*b+c*d+e*f+g*h+i*j (Q1.15*Q1.15=Q2.30)
        APAC    ;acc=a*b+c*d+e*f+g*h
        ADD     j,15             ;acc=a*b+c*d+e*f+g*h+1*j
        .endm

*****
*                               *
*           M A I N   R O U T I N E           *
*                               *
*****

        .text
start:

;~~~~~
;Configure the System Control and Status Registers
;~~~~~
        LDP     #DP_PF1                ;set data page
        SPLK    #0000000011111101b, SCSR1

        LACC    SCSR2                  ;ACC = SCSR2 register
        OR      #0000000000001011b    ;OR in bits to be set
        AND     #0000000000001111b    ;AND out bits to be cleared

        SACL    SCSR2                  ;store to SCSR2 register
;~~~~~
;Disable the watchdog timer

```

```

;~~~~~
LDP    #DP_PF1                ;set data page
SPLK   #0000000011101000b, WDCR

;~~~~~
;Setup external memory interface for LF2407 EVM
;~~~~~
LDP    #temp1                 ;set data page
SPLK   #0000000001000000b, temp1

OUT    temp1, WSGR

;~~~~~
;Setup shared I/O pins
;~~~~~
LDP    #DP_PF2                ;set data page
SPLK   #0000000001000000b, MCRA ;group A pins

SPLK   #1111111000000000b, MCRB ;group B pins

SPLK   #0000000000000000b, MCRC ;group C pins

;~~~~~
;Configure IOPC0 pin as an output
;~~~~~
LDP    #DP_PF2                ;set data page
LACC   #0100h                 ;ACC = 0100h
OR     PCDATDIR                ;OR in PCDATDIR register
SACL   PCDATDIR                ;store result to PCDATDIR

;~~~~~
;Setup the software stack
;~~~~~
stk_len    .set    100                ;stack length
stk        .usect  "stack",stk_len    ;reserve space for stack

LAR        AR1, #stk                  ;AR1 is the stack pointer

;~~~~~
;Setup timers 1 and 2, and the PWM configuration
;~~~~~
LDP    #DP_EVA                 ;set data page
SPLK   #0000h, T1CON            ;disable timer 1
SPLK   #0000h, T2CON            ;disable timer 2

SPLK   #0000000000000000b, GPTCONA

;Timer 1: Configure to clock the PWM on PWM1 pin.
;Symmetric PWM, 20KHz carrier frequency, 25% duty cycle
SPLK   #0000h, T1CNT            ;clear timer counter
SPLK   #pwm_half_per, T1PR      ;set timer period

```

```

SPLK    #0000h, DBTCONA           ;deadband units off
SPLK    #pwm_duty, CMPR1          ;set PWM duty cycle
SPLK    #00000000000000010b, ACTRA
SPLK    #10000010000000000b, COMCONA
SPLK    #00001000010000000b, T1CON

;Timer 2: configure to generate a 250ms periodic interrupt
SPLK    #0000h, T2CNT             ;clear timer counter
SPLK    #timer2_per, T2PR         ;set timer period
SPLK    #11010111010000000b, T2CON

;~~~~~
;Other setup
;~~~~~
;LED index initialization
LDP     #LED_index                ;set data page
SPLK    #1h, LED_index            ;initialize the LED index

;~~~~~
;Setup the core interrupts
;~~~~~
LDP     #0h                      ;set data page
SPLK    #0h,IMR                  ;clear the IMR register
SPLK    #111111b,IFR             ;clear any pending core interrupts
SPLK    #000001b,IMR             ;enable desired core interrupts

;~~~~~
;Setup the event manager interrupts
;~~~~~
LDP     #DP_EVA                  ;set data page
SPLK    #0FFFFh, EVAIFRA         ;clear all EVA group A interrupts
SPLK    #0FFFFh, EVAIFRB        ;clear all EVA group B interrupts
SPLK    #0FFFFh, EVAIFRC        ;clear all EVA group C interrupts
SPLK    #00000h, EVAIMRA         ;enabled desired EVA group A interrupts
SPLK    #00001h, EVAIMRB        ;enabled desired EVA group B interrupts
SPLK    #00000h, EVAIMRC        ;enabled desired EVA group C interrupts

LDP     #DP_EVB                  ;set data page
SPLK    #0FFFFh, EVBIFRA         ;clear all EVB group A interrupts
SPLK    #0FFFFh, EVBIFRB        ;clear all EVB group B interrupts
SPLK    #0FFFFh, EVBIFRC        ;clear all EVB group C interrupts
SPLK    #00000h, EVBIMRA         ;enabled desired EVB group A interrupts
SPLK    #00000h, EVBIMRB        ;enabled desired EVB group B interrupts
SPLK    #00000h, EVBIMRC        ;enabled desired EVB group C interrupts

;~~~~~
;~~~~~
SPI_INIT:
LDP     #6

```

```

;      SETC      INTM      ;Disable interrupts initialization
;      SPLK      #0000h, IMR      ;MASK ALL CORE INTERRUPTS
;      LACC      IFR      ;READ INTERRUPT FLAGS
;      SACL      IFR      ;CLAR ALL INTERRUPT FLAGS
      SPLK      #0h,temp
      OUT        temp,WSGR      ;Set SMIF to run with no wait states.
      CLRC      SXM
      CLRC      OVM
      CLRC      CNF

      LDP        #SPICCR>>7
      SPLK      #004Fh, SPICCR      ;16 CHAR BITS
      SPLK      #000Eh, SPICTL      ;Enable master mode, normal clock
                                      ;and disable talk. WAS 6
      SPLK      #0002h, SPIBRR      ;Set up the SPI to max speed.

      ldp        #MCRB>>7      ;Set up the GPIO pins to function
      SPLK      #003Ch, MCRB      ;as SPI pins

      LDP        #SPICCR>>7
      SPLK      #00CFh, SPICCR      ;Relinquish SPI from Reset. 8F
;~~~~~
;init covariance, state estimate,
      LDP        #6
      SPLK      #21877,p
      SPLK      #6077,p+1
      SPLK      #20230,p+2
      SPLK      #2381,p+3
      SPLK      #3585,p+4
      SPLK      #11404,p+5
      SPLK      #10583,p+6
      SPLK      #4465,p+7
      SPLK      #3211,p+8
      SPLK      #21821,p+9
      SPLK      #4305,p+10
      SPLK      #4755,p+11
      SPLK      #2048,p+12
      SPLK      #1473,p+13
      SPLK      #1350,p+14
      SPLK      #4375,p
      SPLK      #1215,p+1
      SPLK      #4046,p+2
      SPLK      #476,p+3
      SPLK      #717,p+4
      SPLK      #2281,p+5
      SPLK      #2117,p+6
      SPLK      #893,p+7
      SPLK      #642,p+8
      SPLK      #4364,p+9
      SPLK      #861,p+10
      SPLK      #951,p+11

```

```

SPLK    #410,p+12
SPLK    #295,p+13
SPLK    #270,p+14

SPLK    #0,H
SPLK    #0,H+1
SPLK    #0,H+2
SPLK    #0,H+3
SPLK    #0,H+4

SPLK    #15745,MU
SPLK    #12077,MU+1
SPLK    #8690,MU+2
SPLK    #46018,MU+3
SPLK    #16757,MU+4
SPLK    #47758,MU+5
SPLK    #11776,MU+6
SPLK    #51714,MU+7
SPLK    #4869,MU+8
SPLK    #38402,MU+9

SPLK    #16190,MU
SPLK    #14034,MU+1
SPLK    #8258,MU+2
SPLK    #6263,MU+3
SPLK    #8976,MU+4
SPLK    #4371,MU+5
SPLK    #6132,MU+6
SPLK    #29072,MU+7
SPLK    #1273,MU+8
SPLK    #703,MU+9

SPLK    #2802,a1
SPLK    #7698,b1
SPLK    #9011,c1
SPLK    #9830,d1

;      SPLK    #81,pnorm      ;was 327 for pnorm of 50. now pnorm = 75
;      SPLK    #100,lam      ;17,lam

;THIS IS FOR THE CIRCULAR BUFFER INIT.
;      LAR      AR5,#000      ;THIS VALUE IS THE AMOUNT TO DOWNSAMPLE BY
;      LAR      AR3,#(X+200)  ;this is the circular buffer
;      LAR      AR4,#200      ;this is the counter for the circular buffer

;      LAR      AR0,RESULT0    ;put result in ar0 for DAC output

;i am adding the following code to make the DAC work better.

```

```

LAR    AR6,#sample1
LAR    AR7,#sample2

LDP    #DP_PF2
LACC   #110100000000010b      ;sets DAC into 1.024 ref. voltage
;      |||||
*      FEDCBA9876543210      ;MSB should be high (DAC requirement)

```

```

; NOTE TO SELF:
;i can do this in less cycles later
;but im just trying to get this thing to work

```

```

MAR    *,AR6                  ;THIS IS FOR AR POINTER TO SAMPLE1
                                ;for output to the dac.

SACL   *,0,AR6
LAR    ARO,*,AR6

LDP    #SPITXBUF>>7
SAR    ARO,SPITXBUF          ;Write xmit value to SPI Transmit Buffer
LDP    #SPISTS>>7
XMIT_RDY0:
BIT     SPISTS,BIT6          ;Test SPI_INT bit
BCND    XMIT_RDY0, NTC       ;If SPI_INT = 0, then repeat loop

;i.e. wait for the completion of transmission,
LDP     #SPIRXBUF>>7         ;ELSE READ SPIRXBUF
LAR     ARO,SPIRXBUF         ;dummy read to clear SPI_INT flag

```

```

;INITIALIZE ADC REGISTERS
LDP     #DP_PF2              ;load data page
SPLK    #010000000000000b, ADCTRL1 ;RESET ADC MODULE
NOP
SPLK    #0011000000010000b, ADCTRL1 ;TAKE ADC OUT OF RESET
*      |||||
*      FEDCBA9876543210

;      SPLK    #0, MAX_CONV          ;SET UP FOR 1 CONVERSION
SPLK    #0001h, MAX_CONV          ;SET UP FOR 2 CONVERSION
;      SPLK    #0000h, CHSELSEQ1      ;CONVERT CHANNEL 0 ONLY
SPLK    #0010h, CHSELSEQ1        ;CONVERT CHANNEL 0 THEN CANNEL 1

LDP     #DP_PF2
SPLK    #0100011000000000b, ADCTRL2 ;reset for seq1
SPLK    #0000011000000000b, ADCTRL2
SPLK    #0010010000000000b, ADCTRL2 ;START CONVERTING
*      |||||

```



```

*          FEDCBA9876543210

;CHK_EOS1: BIT  ADCTRL2, BIT9    ;WAIT FOR SEQ1 BUSY BIT TO CLEAR
;          BCND   CHK_EOS1, NTC    ;IF TC=0, KEEP LOOPING

;          RPT    #8
;          NOP
;THE CONVERSION RESULTS ARE AVAILABLE
;          LACC   RESULT0
;          ROR
;          ROR
;          ROR
;          ROR
;          ADD    #1100000000000000b    ;SETS THE dac FOR FAST CONVERSION
*          |||||
*          FEDCBA9876543210    ;MSB should be set (DAC requirement)
                                ;IN THE RESULTSn REGS

;          SPLK   #0100011000000000b, ADCTRL2    ;reset for seq1
;          SPK    #0000011000000000b, ADCTRL2
;          SPLK   #0010010000000000b, ADCTRL2    ;START CONVERTING
*          |||||
*          FEDCBA9876543210

;          LAR    AR4, #110h

;~~~~~
;Enable global interrupts
;~~~~~
;          CLRC   INTM    ;enable global interrupts
;~~~~~
;Main loop
;~~~~~

loop:
    NOP
    B      loop    ;branch to loop

*****
*  I N T E R R U P T   S E R V I C E   R O U T I N E S          *
*****
;ADC INTERRUPT for captured results;

adc_isr:

;;GET VALUES FROM ADC

    CLRC   SXM

```

```

LDP      #DP_PF2
LACC     RESULT0
ROR
ROR
ROR
ROR

LDP      #6
SACL     sample1
LT       sample1
MPY      #3300
PAC
SACH     sample1,4

LDP      #DP_PF2

LACC     RESULT1
ROR
ROR
ROR
ROR

LDP      #6
SACL     sample2

SACL     sample2
LT       sample2
MPY      #3300
PAC
SACH     sample2,4

;;;;;; now we have the ADC values so now do RLS
setc     SXM

;; now calculate H where H is
;H=[ya*ya ya*yb -ya -yb 1]      H-> Q1.15
      lacc     sample2,11
      NEG                      ;sample2 is in the acc so -> -sample2
      rol
      SACH     H+3,7           ;H[3]=-sample2  NOTE:: H[2]-> Q1.15

;;NOTE: these next three lines can be moved up to right after
;;the load results1 to save a clock cycle.
      LACC     sample1 ;acc=sample1
      NEG                      ;acc=-sample1
      SACL     H+2,3           ;H[2]=-sample1  NOTE:: H[2]-> Q1.15

      SQRA     sample1 ;sample1^2 is now in the preg
      PAC                      ;load acc with sample1^2
      SACH     H,7             ;store sample1^2 in H[0]

```

[illegible]

```

ADD      MU+2,16
ADDS     MU+3
SACH     MU+2,0          ;MU[1]=MU[1]+K[1]*E
SACL     MU+3,0

MPY      x+2              ;Preg=K[2]*E
PAC                      ;acc=K[2]*E
RPT      #3
SFR
ADD      MU+4,16
ADDS     MU+5
SACH     MU+4,0          ;MU[2]=MU[2]+K[2]*E
SACL     MU+5,0

MPY      x+3              ;Preg=K[3]*E
PAC                      ;acc=K[3]*E
RPT      #3
SFR
ADD      MU+6,16
ADDS     MU+7
SACH     MU+6,0          ;MU[3]=MU[3]+K[3]*E
SACL     MU+7,0

MPY      x+4              ;Preg=K[4]*E
PAC
RPT      #3
SFR                      ;acc=K[4]*E
ADD      MU+8,16
ADDS     MU+9
SACH     MU+8,0          ;MU[4]=MU[4]+K[4]*E
SACL     MU+9,0

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;mu is calculated
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;get a new sample
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
LDP      #DP_PF2
SPLK     #0100011000000000b, ADCTRL2    ;reset for seq1
SPLK     #0000011000000000b, ADCTRL2
SPLK     #0010010000000000b, ADCTRL2    ;START CONVERTING
*        ||||||||||||||
*        FEDCBA9876543210
LDP      #6

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;now estimate a,b,c,d
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;calculate b1*b1

```

```

        SQRA    b1                ;b1^2 is now in the preg Q.26
        PAC                    ;load acc with b1^2
        SACH    temp,3            ;store b1^2 in temp[0] Q.13

;calculate c1

        LT      MU+4              ;LT=MU[2]
        MPY     temp              ;Preg=MU[2]*b1*b1 Q.26
        LTP     a1                ;LT=a1;acc=MU[2]*b1*b1
        ROR                    ;acc=1/2*MU[2]*b1*b1
        NEG                    ;acc=-1/2*MU[2]*b1*b1
        MPY     d1                ;Preg=a1*d1
        LTS     MU+6              ;LT=MU[3],acc=-1/2*MU[2]*b1*b1-a1*d1
        SACH    c1,3              ;c1=-1/2*MU[2]*b1*b1-a1*d1 Q.13

;calculate d1

        MPY     temp              ;Preg=MU[3]*b1*b1 Q.26
        LTP     d1                ;LT=d1;acc=MU[3]*b1*b1
        ROR                    ;acc=1/2*MU[3]*b1*b1
        NEG                    ;acc=-1/2*MU[3]*b1*b1
        MPY     b1                ;Preg=d1*b1;
        LTS     MU+2              ;LT=MU[1];acc=-1/2*MU[3]*b1*b1-d1*b1
        ADD     d1,13             ;acc=d1-1/2*MU[3]*b1*b1-d1*b1
        SACH    d1,3              ;d1=-1/2*MU[3]*b1*b1-d1*b1 Q.13

;calculate a1

        MPY     temp              ;Preg=MU[1]*b1*b1 Q.26
        LTP     a1                ;LT=a1;acc=MU[1]*b1*b1
        ROR                    ;acc=1/2*MU[1]*b1*b1
        MPY     b1                ;Preg=a1*b1
        LTS     temp              ;LT=temp,acc=1/2*MU[1]*b1*b1-a1*b1
        ADD     a1,13             ;acc=a1+1/2*MU[1]*b1*b1-a1*b1
        SACH    temp,3            ;temp=a1+1/2*MU[1]*b1*b1-a1*b1 Q.13

;calculate b1

        MPY     MU                ;Preg=MU[0]*b1*b1 Q.26
        LTP     a1                ;LT=a1,acc=MU[0]*b1*b1
        NEG                    ;acc=-MU[0]*b1*b1
        MPY     a1                ;Preg=a1*a1
        APAC                    ;acc=a1*a1-MU[0]*b1*b1
        ROR                    ;acc=1/2*a1*a1-1/2*MU[0]*b1*b1
        ADD     #4096,13          ;acc=1/2+1/2*a1*a1-1/2*MU[0]*b1*b1
        ADD     b1,13             ;acc=b1+1/2+1/2*a1*a1-1/2*MU[0]*b1*b1
        SACH    b1,3              ;b1=b1+1/2+1/2*a1*a1-1/2*MU[0]*b1*b1 Q.13

;save temp to a1 for new update.
        LACC     temp              ;acc=a1+1/2*MU[1]*b1*b1-a1*b1 Q.13
        SACL     a1                ;a1=a1+1/2*MU[1]*b1*b1-a1*b1 Q.13

```

```

;;;NOTE: This next part can be commented out if the output of output2
;;; can have an offset of c1 instead of d1
;;now find inv(b1) (16 bit inverse)
;qinv4:
;      SETC      OVM                      ;To saturate ABS value of 0x8000 to 0x7fff

;;      SETC      SXM
;;      SPM       #0
;;      LAR       AR2,#temp              ; AR0->temp
;;      MAR       *,AR2
;
;=====
;; From here the code is same as CcA
;=====
;;
;      LACC      b1,16                  ; Load the input to the ACCH
;
;      BCND      calculate3,NEQ        ; If input is zero, return with 0
;      LACC      #0000h
;      B         return3
;
;calculate3:
;      BCND      positive3, GT
;      ABS                      ; Obtain the absolute value of x<0
;
;positive3:
;      SACH      *                      ; Store |x| in 16.0
;      LACC      #01h,15              ; Load 1 in Q15
;
;      RPT       #15
;      SUBC      *                      ; Perform division to obtain Q
;
;      AND       #0ffffh
;
;      SACL      *
;      BIT       *,0,AR2
;      BCND      isnot3,NTC
;      LACC      #7ffffh
;
;isnot3:
;      BIT       b1,0
;      BCND      return3,NTC
;      NEG
;
;return3:
;      CLRC      OVM                  ; the INV is in the acc
;
;      SACL      dhat
;
;;;Dhat is

```



```

        BCND    calculate,NEQ    ; If input is zero, return with 0
        LACC    #0000h
        B       return

calculate:
        BCND    positive, GT
        ABS                                ; Obtain the absolute value of x<0

positive:
        SACH    *                    ; Store |x| in 16.0
        LACC    #01h,15              ; Load 1 in Q15

        RPT     #15
        SUBC    *                    ; Perform division to obtain Q

        AND     #0ffffh

        SACL    *
        BIT     *,0,AR2
        BCND    isnot,NTC
        LACC    #7ffffh

isnot:   BIT     R,0
        BCND    return,NTC
        NEG

return:   CLRC    OVM                ; the INV is in the acc
        SACL    R

;calculate two times the freq
        SQRA    temp1+1              ;Preg=temp[0]^2
        LACC    #0000h              ;acc=0
        SQRS    temp1                ;acc=temp[0]^2; Preg=temp[1]^2
        APAC                                ;acc=temp[0]^2+temp[1]^2
        SACH    timestwo,1           ;R=-temp[0]^2+temp[1]^2

        MPY     temp1+1              ;Preg=temp[0]*temp[1]
        PAC
        SACH    timestwo+1,2         ;acc=2*temp[0]*temp[1]

        LT      R                    ;LT=inv(R)
        MPY     timestwo              ;Preg=timestwo[0]*inv(R)
        PAC                                ;acc=timestwo[0]*inv(R)
        ror
        SACL    timestwo,0           ;timestwo=timestwo[0]*inv(R)

        MPY     timestwo+1           ;preg=timestwo[1]*inv(R)
        PAC                                ;ACC=timestwo[1]*inv(R)
        ror
        SACL    timestwo+1,0         ;timestwo+1=timestwo[1]*inv(R)

```



```

;now calculate threetimes
    LT      timestwo      ;LT=timestwo[0]
    MPY      temp1         ;Preg=timestwo[0]*temp[0]
    LTP      timestwo+1    ;acc=timestwo[0]*temp[0]
    MPY      temp1+1       ;Preg=timestwo[1]*temp[1]
    SPAC                     ;acc=thimestwo[0]*temp[0]
                        ;-timestwo[1]*temp[1]
    SACH      threetimes,2 ;threetimes[0]=thimestwo[0]*temp[0]
                        ;-timestwo[1]*temp[1]

    MPY      temp1         ;Preg=timestwo[1]*temp[0]
    LTP      timestwo      ;acc=timestwo[1]*temp[0]
    MPY      temp1+1       ;Preg=timestwo[0]*temp[1]
    APAC                     ;acc=timestwo[1]*temp[0]
                        ;+timestwo[0]*temp[1]
    SACH      threetimes+1,2 ;threetimes[1]=timestwo[1]*temp[0]
                        ;+timestwo[0]*temp[1]

;now the last step OUTPUT DATA

    LACC      threetimes    ;acc=threetimes[0]
    sub       c1,2          ;acc=threetimes[0]+c1
    ROR
    ROR
    ROR
    SACL      out1

    LACC      threetimes+1  ;acc=threetimes[1]
    sub       c1,2          ;acc=threetimes[1]+c1
    ROR
    ROR
    ROR
    SACL      out2

;THIS IS A CIRCULAR BUFFER. X[k]is the circular buffer for debug mode.
;if (k==0){
;    k=160;
;}
;if(n>80){
;    n=0;
;X[k]=H[3];
;k=k+1;
;}
;n=n+1;

;    MAR      *,AR5
;    BANZ     NEXXT1
;    LAR      AR5,#0        ;THIS VALUE IS THE AMOUNT TO DOWNSAMPLE BY

;    MAR      *,AR4
;    BANZ     NEXXT2

```

```

;      LAR      AR3,#(X+200)
;      LAR      AR4,#400
;      B        END22
;NEXT2:
;      MAR      *,AR4
;      MAR      *-,AR3
;      LACC     p+3
;      SACL     *-
;END22:

;      B        END11
;NEXT1:      MAR      *,AR5
;      MAR      *-

;END11:

```

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;i added this on 5/27/04
      CLRC      SXM
      LACC      out1
      and       #00FFFh
      ADD       #1100000000000000b      ;SETS THE dac FOR FAST CONVERSION
*          |||
*          FEDCBA9876543210      ;MSB should be set (DAC requirement)
      SACL      out11

      LACC      out2
      and       #00FFFh
      ADD       #0101000000000000b      ;SETS THE dac FOR FAST CONVERSION
*          |||
*          FEDCBA9876543210      ;MSB should be set (DAC requirement)
      SACL      out22

;;;;;;;;;;;;;;;;;;

      LDP       #SPITXBUF>>7
      SACL      SPITXBUF      ;Write xmit value to SPI Transmit Buffer
      LDP       #SPISTS>>7
XMIT_RDY1:
      BIT       SPISTS,BIT6      ;Test SPI_INT bit
      BCND      XMIT_RDY1, NTC      ;If SPI_INT = 0, then repeat loop

;i.e. wait for the completion of transmission,
      LDP       #SPIRXBUF>>7      ;ELSE READ SPIRXBUF
      LAR       ARO,SPIRXBUF      ;dummy read to clear SPI_INT flag

```

```

        LDP      #6
        LACC     out11
        LDP      #SPIRXBUF>>7          ;ELSE READ SPIRXBUF
        SACL     SPITXBU                ;Write xmit value to SPI Transmit Buffer
        LDP      #SPISTS>>7

XMIT_RDY2:
        BIT      SPISTS,BIT6            ;Test SPI_INT bit
        BCND     XMIT_RDY2, NTC         ;If SPI_INT = 0, then repeat loop

;i.e. wait for the completion of transmission,
        LDP      #SPIRXBUF>>7          ;ELSE READ SPIRXBUF
        LAR      ARO,SPIRXBUF          ;dummy read to clear SPI_INT flag

        CLRC     INTM                  ;re-enable interrupts
        RET      ;return from the interrupt

;~~~~~
;GP Timer 2 period interrupt (core interrupt INT3)
;~~~~~

timer2_isr:

;Context save to the software stack
        MAR      *,AR1                ;ARP=stack pointer
        MAR      ++                  ;skip one stack location (required)
        SST      #1, ++              ;save ST1
        SST      #0, ++              ;save ST0
        SACH     ++                  ;save ACCH
        SACL     ++                  ;save ACCL

;Clear the T2PINT interrupt flag
        LDP      #DP_EVA              ;set data page
        SPLK     #00001h, EVAIFRB     ;clear T2PINT flag

;Sequence the LED bank on the LF2407 EVM
        LDP      #LED_index           ;set data page
        OUT      LED_index, LED       ;light the LED
        LACC     LED_index,1          ;load LED index with left shift of 1
        SACL     LED_index            ;store updated index
        SUB      #0010h              ;subtract the mask
        BCND     done, LT             ;branch if index not ready for reset
        SPLK     #1h, LED_index       ;reset LED index to 1
done:

;Toggle the IOPC0 pin
        LDP      #DP_PF2              ;set data page
        LACC     #0001h              ;ACC = 0001h
        XOR      PCDATDIR             ;XOR the IOPC0 bit to toggle the pin
        SACL     PCDATDIR             ;store result to PCDATDIR

```

```
;context restore from the software stack
MAR    *, AR1          ;ARP = AR1
MAR    *-              ;SP points to last entry
LACL   *-              ;restore ACCL
ADD     *-,16           ;restore ACCH
LST     #0, *-          ;restore ST0
LST     #1, *-          ;restore ST1, unskip one stack location
CLRC    INTM            ;re-enable interrupts
RET                     ;return from the interrupt
```

APPENDIX C

MATLAB CODE

Mirror Estimate

```

clear;

load 'inout0.txt';

x=inout0';

function er = func(mu,y1,y2)
    er = y2 - (mu(3)*(3*(y1-mu(2))/mu(1) - 4*( (y1-mu(2))/mu(1)).^3) + mu(4));
end;

ind=1:size(x,1);

y1 = x(ind,1);
y2 = x(ind,2);

mu0=[1 1 1 1]';

[mu,info,iter]=lmsvd('func',mu0,[],y1,y2);

y2hat = -func(mu,y1,0);

figure(1);
ind2 = 23000:27000;
t2 = ind2 * (1/100000);

plot(t2, y2(ind2),'rx', t2, y2hat(ind2));

ylabel('DSP output and modeled DSP output');
xlabel('time (sec)');

figure(2);
hist(y2-y2hat,100,1.0);
ylabel('Error distribution for DSP output vs. model DSP output.');
```

axis([-0.15 0.15 0 2.5e5])

Bootstrap Code

```

clear all;
tic
```

```

N = 20000; %# number of distributions to try

% angles
onerev = [0 pi/3 2*pi/3 pi -2*pi/3 -pi/3]';
theta = [];

for i=1:12
    theta = [theta;onerev];
end;

% angle vector
phi=[];
for i=1:length(theta)
    phi = [phi; ones(100,1)*theta(i)];
end;

r = .69;
a = .83;
b = 1.07;
d = 1.39;
ph = -110/180*pi;
sigma = 0.01;
sigma2 = .01;
SNR=(r*d)^2/2/(sigma2^2);
sigma1 = sqrt((r/2)/SNR);

% bootstraps are in ROWs of err, angles are in subsequent columns.
% complex representation of angle is stored in uh
%
err = zeros(N,length(theta));
uh = err;

% boots.
for i = 1:size(err,1)

    n1 = sigma1*randn(size(phi));
    n2 = sigma2*randn(size(phi));
    la = r*cos(phi) + a + n1;
    lb = r*d*cos(phi+ph) + b + n2;

    % theta's
    for k = 1:length(theta)
        [u,thetahat] = estimate(la(1:k*100),lb(1:k*100));

        % this error is between -pi, pi

```

```

err(i,k) = angle( u / (j*sin(theta(k))+cos(theta(k))));

uh(i,k) = u;

% compute angle error additive noise (use same noise as for estimate above)

q = j*(sin(theta(k))+n2(k*100)) + cos(theta(k))+n1(k*100);
q = q/norm(q);

ierr(i,k) = angle( q / (j*sin(theta(k))+cos(theta(k))));
end;
end;

toc

mse = mean(err.*err);
me = mean(err);
imse= mean(ierr.*ierr);
ime = mean(ierr);
figure(1);
plot(unwrap(theta),log10(mse),unwrap(theta),...
     log10(ones(size(theta))*sigma1*sigma2),...
     unwrap(theta),log10(ones(size(theta))*sigma1*sigma2),'-');

axis([0 74.35 -5 1])
%title('Mean squared error of  $\hat{\phi}$ ','interpreter','latex');
xlabel('Mirror angle  $\phi$ ,  $^{\circ}$ ','interpreter','latex');
ylabel('Variance  $\log_{10}(\hat{\phi})$ ','interpreter','latex');

figure(2);

[n,m]=size(uh);
m=52;
n=m;
uhh = uh(:,n:m);
ieh = ierr(:,n:m);
erh = err(:,n:m);

nn=linspace(-.076,.076,250);
[xx,nn]=hist(erh(:),nn);
[yy,nn]=hist(ieh(:),nn);
xx=xx/max(xx);
yy=yy/max(yy);
[xs,ys]=stairs(nn,yy);
[xs1,ys1]=stairs(nn,xx);
axis([min(nn) max(nn) 0 .02]);
plot(xs,ys,xs1,ys1);
%title('Bootstrap estimate');
xlabel('Error distribution for  $\hat{\phi}$ ,  $^{\circ}$ ','interpreter','latex')

```



```
ylabel('Density function');
```

Closed Loop Test Code

```
clear all
%load scope data
load innew0.txt;
ch1=innew0(1,:);
ch2=innew0(2,:);

step = [6960      16960
        25900     35900
        44600     54600
        63560     73560
        82200     92200
       101200    111200
       119840    129840
       138780    148780
       157420    167420
       176450    186450
       195070    205070
       214000    224000
       232690    242690
       251600    261600
       270300    280300
       289310    299310
       307970    317970
       326900    336900
       345560    355560
       364580    374580
       383200    393200
       402210    412210
       420830    430830
       439710    449710
       458400    468400
       477390    487390
       496060    506060
       515020    525020
       533690    543690
       552680    562680
       571330    581330
       590270    600270
       608900    618900
       627910    637910
       646570    656570
       665520    675520
       684170    694170
       703150    713150]
```

```

        721820      731820
        740790      750790
        759430      769430];
% create an index and angle vector.

ind=[];
theta=[];

for i=1:size(step,1)
    chk = (step(i,1):step(i,2))';
    ind=[ind; chk];
    theta = [theta; 2*pi*mod(i,6)/6 * ones(size(chk))];
end;

% determine a phase, DC offset, and amplitude for each scope
% signal, given the fixed angle increment. All parameters are
% independent on each channel.
%
A = [ones(size(theta)) cos(theta) sin(theta)];
mu1 = A \ ch1(ind);
mu2 = A \ ch2(ind);
plot(ind,ch1(ind) - A*mu1);
plot(ind,ch2(ind) - A*mu2, 'r');
ch1h = A*mu1;
ch2h = A*mu2;

err = angle((ch1(ind)+sqrt(-1)*ch2(ind)) ./ (ch1h + sqrt(-1)*ch2h));

ec1 = ch1(ind)-ch1h;
ec2 = ch2(ind)-ch2h;

bin=linspace(-.15,.15,20);

vec1=A*mu1;
vec2=A*mu2;
figure(1)
%MAKE Ellipse
A1=[ones(1000,1) cos(linspace(0,2*pi,1000))' sin(linspace(0,2*pi,1000))'];
plot(ch1(ind),ch2(ind), 'g.', A*mu1, A*mu2, 'r.')
hold on
%MAKE CROSS ARROWS
for i=1:12
    plot([vec1(step(i))-0.108; vec1(step(i))+0.108],...
        [vec2(step(i)) ;vec2(step(i))], 'r')
end
for i=1:12

```

```

plot([vec1(step(i)); vec1(step(i))],...
     [vec2(step(i))-.15 ;vec2(step(i))+.15], 'r')
end
%plot distribution
plot(A1*mu1,A1*mu2,'b--')
axis([-0.25 2.5 -0.25 2.5])
xlabel('$y_1$ (volts)','interpreter','latex')
ylabel('$y_2$ (volts)','interpreter','latex')
hold off

```