Dynamic Structural Shape Estimation Using Integral Sensor Arrays

by

Peter Stewart Lively

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Feb 2000

Author .....
.....
Department of Aeronautics and Astronautics
January 28, 1999

Certified by.....
Nesbitt W. Hagood
Professor
Thesis Supervisor

Certified by..
.....
Mauro J. Atalla
Research Associate
Thesis Supervisor

Accepted by.....
.....
Nesbitt W. Hagood
Chairman, Department Committee on Graduate Students

Dynamic Structural Shape Estimation Using Integral Sensor Arrays

by

Peter Stewart Lively

Submitted to the Department of Aeronautics and Astronautics
on January 28, 1999, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Modern structures are becoming intelligent structures, with embedded sensors and actuators. This is particularly true for aircraft where any extra weight is costly. Shape estimation is one technique that can utilize embedded sensors to improve the aircrafts performance. Having shape knowledge allows for embedding phased antennae arrays into large structural surfaces such as wings. The shape information is used to adjust the phasing for optimal signaling. Another application is for HALE aircraft, where the flexible wings are vulnerable to gusts. Shape estimation allows a controller to steer through the gust to avoid failure.

Currently the most robust method of shape estimation utilizes global shapes, which are generally the lowest structural modes. These modes produce the greatest levels of deflection. The sensor readings, usually strain gages, are passed through a weighting matrix which provides an estimate of the modal amplitudes. Summing the amplitudes provides and estimate of the total shape. This has been shown to provide accurate results for static structures. Current dynamic shape estimation techniques use the static technique for each time step. The aliasing of the higher modes, which is largely not seen in the static case, occurs strongly in the dynamic case. Higher frequency modes cause high strain with relatively little displacement. In many cases the aliasing produces signal to noise ratios significantly greater than unity.

This paper presents a new technique that uses the dynamic model of the structure to greatly reduce the effects of aliasing. The premise is to use a Kalman filter to sift out the desired low frequency modes and treat the higher modes as part of the noise. Also, unlike the static techniques, the Kalman filter allows sensing of more modes than there are sensors and takes into account the strain gauge errors. Numerical analysis shows that the Kalman filtering technique can reduce the error from a thousand percent down to less than a percent in an ideal case. Including sensor noise and modeling error reduces the efficacy of the method but still produces a significant improvement.

Thesis Supervisor: Nesbitt W. Hagood
Title: Professor

Thesis Supervisor: Mauro J. Atalla
Title: Research Associate

Acknowledgments

A number of people helped me get my experiment off the ground, and they most of all should be thanked. So thanks Mauro, Michael, Cagri.

But most of all I would like to thank the people that helped me the last few weeks, even if they didn't know it. Thanks Michael, John, Mary, Will, John and my little Dutch friend.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Intelligent Structures

Intelligent structures are being used increasingly for hi-tech projects particularly for aerospace vehicles and applications [25, 28]. An intelligent structure utilizes integral sensors and actuators to increase performance. These integral systems can either augment or replace their on-board counterpart.

Warkentin et al. [33, 34, 35] worked extensively on embedding electronic components into a composite structure and modeling the resulting structure. Of particular importance is embedding piezoelectric actuators and correctly predicting their behavior. Their work on modeling of the piezoelectric materials allows for the piezoelectric materials to be used efficiently, and to develop algorithms for optimal placement for control applications.

John Rodgers, et al. [29] have used embedded Active Fiber Composites (AFC) in the construction of an active twist rotor blade. The active rotor blade has the advantage of high bandwidth actuation allowing for control of higher modes. These blades will be able to reduce the noise produced by helicopters, increasing performance and comfort levels. Another benefit is that it allows for simplification of the hub machinery, increasing life of the helicopter as a whole.

Embedded arrays of PZT's were used by Mark Lim and Fu Kuo Chang [30, 24] to create a method of damage detection. The PZT's act as both actuators and

sensors, where one of the PZT's is used to send out a high frequency strain burst that is picked up by the other embedded PZT's. The undamaged baseline is compared to subsequent readings. Increases in time lag and scatter indicate damage in the structure. Comparing the various lag readings and levels of scatter the damage can be located within the structure.

1.2 Modal Detection and Sensing

For many control applications modal detection is important. Lee and Moon [23] have developed a method for shaping sensors to sense one particular mode. The spatial shape of the sensor is determined to provide a signal proportional to the desired mode. This shape is also created to be orthogonal to the undesired modes. Burke and Hubbard [7] also developed similar ideas that are used for modal control of a beam. Unfortunately this requires very large sensors that take up most of the structural area.

For many structures Vári and Heyns [32] have shown that the strain shapes are not necessarily orthogonal. The total deformation energies of the structure maintain their orthogonality, but that includes deformations at the boundaries which cannot be included in any simple manner.

Miller et. al. [26] developed shaped sensors that are local rather than distributed. The spatial shape of the sensor detects spatial waves of a certain length. The length of the spatial wave corresponds to a structural frequency, with longer waves corresponding to lower frequencies. The size and shape of the sensor determines the roll-off point and rate respectively. These shaped sensors are very useful to reduce the sensitivity to high-frequency modes. These are particularly useful for long slender applications, such as beams and rods where the spatial sensitivity of the sensor can be controlled by adjusting the transverse width. The transverse sensitivity of the sensors makes this much more difficult on a plate structure.

Han and Lee [16] used an array of piezoelectric sensors and actuators to control the deflections of a cantilevered plate. The plate was covered with piezoelectric patches

and the system was identified. Boolean weightings were applied to each of the sensor patches to best isolate the first mode of the structure, and reduce the observability of the higher modes. A similar weighting system was applied to the actuator piezos. The resulting configuration showed very good performance at reducing the controlled mode without detriment to the higher modes. This approach has a number of shortcomings however. The sensors layout once chosen was hardwired in, and any changes to the structure would dramatically alter performance.

Michael Fripp [13] has investigated the use of an array piezoelectric sensors with continuous weightings for vibration and structural-acoustic control. The weightings are chosen to correspond to the modes of interest, which are to be controlled. The sensor weightings are determined from the sensor transfer functions and can be updated with changes in the structural response.

1.3 Shape Estimation

The need for Shape Estimation in an intelligent structure is becoming more apparent. GPS booms, for example, require very precise alignment where a 0.1 degree error in alignment can cause a 0.5 mile ground error [21]. Another precision shape application is for conformal antenna arrays for high performance aircraft. The antenna is built into the wing of the aircraft and requires very precise alignment[20].

Shape estimation draws upon various aspects of intelligent structures, primarily the use of arrays of sensors. The larger the array of sensors the better the estimate becomes. Additionally, shape estimation can utilize sensors already in place for other purposes, such as strain monitoring, or acoustic damping.

The basis for most higher-order shape estimation techniques is to map displacement fields to strain fields. Foss and Haugse [12] used large arrays of strain gages and accelerometers to map strain shapes and the corresponding deflection shapes. The testbed was a cantilevered plate. They isolated individual modes and used curve fitting techniques to interpolate between sensor locations. The final result is accurate strain shape maps and displacement mode shapes.

Mark Andersson [2] investigated the use of arrays of shaped sensors for shape estimation. His work included comparisons of integration techniques to modal sensing techniques, as well as the effects of shaped sensors and the number of sensors on these results. The results varied depending on the scheme used and the sensor, but the trend was that a larger number of sensors produced more favorable results, and that shaping the sensors had marginal effects in most cases.

Not all shape estimation methods use a strictly strain gage approach though. Baz and Poh [4] developed a wire gage that does point estimations of the slope of the beam through integration of the strain down the length of the wire. Their work included using an oscillating platform to actuate a beam that is clamped to it. This allows them to excite the beam at specific frequencies. Their results were rather accurate, but they only conducted tests at low frequencies.

The major shortcoming with the shape estimation performed to date is that little has been done for a vibrating structure. Kirby et. al. [21] performed experiments on a structure undergoing a step loading, using strain integration techniques. The transient results proved accurate, but there was a high level of error immediately after the step load that is unaccounted for analytically. Much of this error is undoubtedly from the aliasing effects of the high frequency modes that are not accounted for with standard static shape estimation techniques.

The purpose of this work is to develop a strategy to address the aspect of dynamics in shape estimation. To do this modal estimation techniques developed for controls applications are applied to current shape estimation techniques. Sensor weighting functions are the most basic techniques, and are already being utilized in the global shape estimation method. More advanced methods filter the signal in frequency, as the weighted sensors are designed to filter the response spatially.

Modal estimation techniques thus provide a better estimate of the modal amplitudes, that are then combined with the mode shapes to render a final deflection shape.

1.4 Design Requirements

In designing a dynamic shape estimation method it is important to recognize and design toward certain performance and functional requirements. A methodology that works poorly is worse than not having one at all.

The primary performance requirement for a shape estimation method is that the error in the estimated shape must be less than the actual deflection levels. Any method that produces greater error than the deflection level is worse than assuming no deflection at all.

In order to reduce the errors in the system, the method must be able to account for four different types of errors:

1. Aliasing of higher modes
2. Truncation of the model
3. Sensor error
4. Model error

The importance of these errors is dependent on the number of sensors, sensed modes, bandwidth of the external forcing and model accuracy.

Aliasing errors occur for two reasons in a dynamic system. The primary reason is that the sensor weighting matrix is only capable of differentiating between as many modes as there are sensors. Higher modes are thus interpreted as combinations of the lower modes, often producing very large errors. The second form the aliasing takes is digital aliasing due to the rate of sensing. If the bandwidth of the sensing is suitably high, this does not have much effect however.

Model truncation is, in a way, the opposite of aliasing. Model truncation occurs because not all of the structural modes can be added to the total deflection. The estimated deflection reflects the truncation by not having those modes appear in the estimate. This means that the estimated deflection will necessarily be simpler than the real deflection. The effects of model truncation are not as severe for shape estimation as they are for controls, since deflection rolls-off very quickly with frequency.

Sensor errors occur in many different forms for shape estimation. All sensors have

an intrinsic level of error, due to sensitivity limitations, line noise, drift, etc. Beyond that, there is also error due to misalignment and incorrect placement of the sensor. Small errors in the location of the sensor can have profound impact on their reading especially for higher modes that have short spatial wavelengths.

Model error is the last of the main causes of errors in the system. No matter how sophisticated the finite element package, or the identification software, the structure is not going to be modeled entirely correctly. This is especially true for a structure that is subjected to changes in environment, i.e. hot to cold, or operates in a rugged environment where it will be damaged, since damage changes the structural response. For this problem the two main sources of model error are errors in modal frequencies and physical mode shape.

In addition to having to contend with the various sources of error, the technique must also be fast, a restriction that does not exist for static shape estimation. The dynamic technique should be capable of updating the estimated shape as quickly as new information is available. At the minimum the system rate should be less than the Nyquist frequency for the highest desired mode ($\omega_{max}/2$).

Additional requirements on the performance and functioning can be made as necessary for the specific applications. In general the above and an absolute tolerance on the deflection error at any point suffice to delineate the problem.

The next chapter will go into more detail on the various methodologies that are available for shape estimation, including discussion on the relative merits of each method. Following the methodologies chapter is a chapter on the computer simulations that were run using the various temporal filters. The experimental results are presented after the computer simulations.

Chapter 2

Methodologies

2.1 Introduction

Static shape estimation is the basis for the dynamic methods used in this work. The most basic shape estimation method is simply iterating the static method, while the more complex methods fit dynamic modes to the sensor readings.

In tailoring a static shape estimation method to a specific structural application, a number of parameters must be determined. One must first determine the number, the type, and the locations of the sensors. It is also important to consider whether the number and location of sensors is a fixed parameter or a design parameter.

No matter what method is used, shape estimation starts with some type of model of the structure. This creates a way to relate the readings of the sensors to what is physically happening in the structure. For the simpler methods all that is required is the geometry of the structure. As the methods get more complicated the necessary complexity of the model also increases.

When selecting a methodology for shape estimation the first step is to examine the potential errors, modeling errors, sensing errors, aliasing, etc. Of the possible errors, the presense and potential degree of aliasing is the easiest to predict and design for.

Based on the types and estimated levels of errors, an underlying static method can be chosen. The static method determines how the sensor information is used spatially. After choosing a static method, a dynamic method can be added. The

dynamic method uses temporal information to improve the spatial information in the static method.

2.2 Aliasing

As the leading cause of errors in a shape estimation method it is important to understand exactly what causes the aliasing and how great its effect really is. The manner and degree that aliasing affects the sensors will help determine which method should be used.

Aliasing occurs when one mode is mistaken for another mode, or combination of other modes. In the case of shape estimation this means that the strain produced by an unsensed mode is aliased to the strain from the sensed modes. This produces erroneous readings for the sensed modes causing errors in the estimated shape.

In a structure with few sensors and high bandwidth noise, aliasing can be disastrous, easily causing errors that completely mask the underlying signal. Each additional sensor decreases the error significantly. A lower bandwidth for the noise means that the higher modes do not get excited as much, and are less able to cause aliasing.

Mathematically the cause of aliasing can be demonstrated by examining a vibrating beam. Equation 2.1 shows the governing differential equation for a vibrating beam. For an infinite beam, the transfer function corresponds to the dereverberated transfer function of a generic beam, as shown in Equation 2.2, where the structural wavelength $\lambda \propto 1/\sqrt{\omega}$.

$$\rho\ddot{z} + EIz^{IV} = F(x,t) \tag{2.1}$$

$$z(x,t) = sin(\omega t + \frac{2\pi}{\lambda}) \tag{2.2}$$

This results in a curvature that is proportional to $1/\omega$. This means the curvature to displacement transfer function is proportional to:

$$\left\| \frac{x(\omega)}{x''(\omega)} \right\| = \omega \tag{2.3}$$

Since the strain gage reading is directly proportional to the curvature, this means that the sensor to displacement function rolls up with frequency, which is very undesirable from a controls point of view. The results of this is that the higher modes have proportionally greater influence on the sensed deflection than the lower modes.

2.3 Static Methods

2.3.1 Introduction

Static Methods are the shape estimation techniques that are used for a structure undergoing static deformation, or very low frequency deformation. Static methods are also the first step in creating a dynamic shape estimation method.

2.3.2 Sensors

A strain gage attached to the surface of a bending structure will respond linearly with local curvature. Ideally a pair of strain gages are utilized on opposing sides of the structure. The use of two strain gages allows for compensation for elongation of the structure. Mid-plane deformation produces strain on both sides of the structure with the same sign, whereas curvatures produces strains of opposing signs.

Typically in a thin structure the strains due to curvature will be much greater than those due to mid-plane deformation. This is especially true for beam-like structures which have large transverse loads and small, if any, loads down the length of the structure. This means that a single strain gage on one side of the structure will usually suffice.

2.3.3 Shaped Sensors

A strain gage is not a point sensor. The output signal is in reality the integration of the strain over its entire area. Typically this is not a factor in the use of a strain gage, although it may help determine a size requirement.

Using piezo-electric material it is possible to specifically shape the strain gage's area to improve sensing. This is usually done in one of two ways, global strain gages and discrete strain gages.

Global Strain Gages

Global strain gages are used to pick out a single mode in a deformed structure. The gage is shaped in such a way that the output of the sensor is proportional to the deformation for that mode. The gage is also shaped so as to be orthogonal to the other modes of the structure.

Designing a Global Strain Gage starts with either a very detailed computer model of the structure, or with a detailed physical identification of the structure to be sensored. The next step is to look for regions that have high strain for the desired modes, and low strains for the other modes. To complicate the matter the sensor should be designed to be entirely contiguous. Charette and Berry [8], and Lee and Moon [23] investigated ways to accomplish this. The disadvantage of this method is that the sensor must often be subsequently tailored to account for the sensors weight and stiffness which alter the structural modes. Additionally only a limited number of such sensors are capable of being used on a structure at one time since they cover such large areas.

Discrete Shaped Strain Sensors

A discrete strain gage works on a fundamentally different idea than a globally shaped strain gage. A discrete strain gage is designed to act as a spatial filter for strain waves. The discrete gages cannot be used by themselves to determine deflection, but must be coupled with another shape estimation method. The purpose of the discrete gage is to help filter out the higher modes of the structure.

Discrete shaped strain gages are typically used in a beam structure where most of the strain is in the same direction. This simplified structure allows the assumption that the strain field is invariant across the width of the beam and varies only along the length of the structure. This means that the width of the gage can be used to

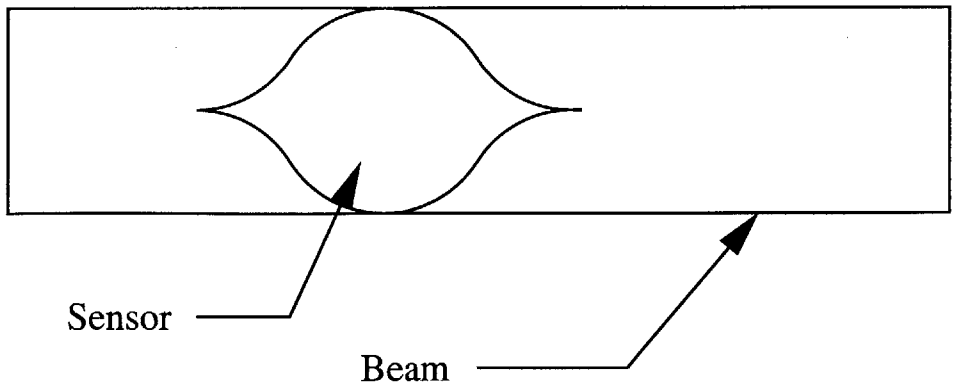


Figure 2-1: Typical Example of a Discrete Shaped Sensor

determine the magnitude of the gage's response at any point along the length of the gage.

Typical discrete gages are shaped in the same way the time signatures for low-pass filters are designed. The idea is for long strain waves to give strong signals and for shorter strain waves to be attenuated. The shorter strain waves correspond to higher frequency modes which are typically undesirable for shape estimation, whereas the longer modes dominate the deflection. A typical example of a spatially filtering strain gage is illustrated in Figure 2-1.

On two dimensional structures the width of the gage is not a feasible design parameter since piezoelectric materials, which is the typical sensor type used for shaped sensors, have high transverse sensitivities. However, even though a beam structure is the easiest for which to design a discrete filter, other structures are capable of utilizing shaped sensors. A circular gage for instance increases the roll-off of the strain gage, compared to a rectangular gage, from $1/\omega$ to $1/\omega^2$ [2].

2.3.4 Integration Methods

The most straight forward class of the shape estimation methods is integration techniques. Since the strain gages mounted on the surface of the structure measure the local curvature, the curvatures can be integrated to determine the deflection.

Almost any integration method can be used for shape estimation. Higher order methods tend to produce better results however. Gaussian integration for instance

effectively doubles the number of strain gages through careful placement of those sensors to be orthogonal to higher modes that do not contribute to deflection. Even without being able to choose the location of the sensors, integrations techniques can be very powerful.

There are however a number of drawbacks. The first is that integration techniques are very difficult to use on complicated structures; their usefulness is basically limited to beams and very simple plate structures. An additional difficulty, specific to using integration methods for dynamic shape estimation, is that the better integration methods are computationally expensive relative to other methods [3].

2.3.5 Global Shape Estimation

Global Shape Estimation is basically the opposite of integration methods. Global Shape Estimation starts with estimated shapes and differentiates them twice to find the curvatures. A matrix correlating the curvatures, which are also the strain gage readings, to the global deformation shapes can thus be created. Inverting this matrix gives the sensing matrix.

Multiplication of the sensing matrix by the sensor readings directly produces the modal amplitudes. Since each mode has a predetermined mode shape corresponding to it, the modal amplitudes can then be multiplied by the displacement curves for their respective mode. Summing the resulting curves gives an estimate of the shape of the structure.

Global Shape Estimation has a number of advantages over the other methods, especially for structures undergoing dynamic deformations. Since the shapes are predetermined, a number of other aspects of the structure can be identified with the same sensors and mode shapes, including information such as stress levels, and slopes which are useful for pointing applications. By examining consecutive time steps, accelerations and velocities can also be determined directly.

Global Shape Estimation also has the advantage of being very easy to adapt to different structures. Once the model of the structure is created and mode shapes are determined, the sensing matrix can be formed, and then the system is ready. For a

dynamic structure the model can even be created by identifying the structure [12].

In terms of suitability for dynamic shape estimation, the global methods are also very good. The time required for the matrix multiplications is very small compared to the calculations required by some of the integration methods.

The primary disadvantage of the sensing global shapes is that there are no innate anti-aliasing features. The complexity of the deformations that can be sensed is proportional to the number of sensors. More specifically the number of mode shapes that can be distinguished is equal to the number of sensors.

2.4 Dynamic Methods

2.4.1 Selection of Base Method

In order to compare accurately the results of different dynamic methods, it was decided that a single base static method should be used. Global Shape Estimation was concluded to be the best choice. The global shapes that the methods will utilize are the deflection mode shapes of the structure.

There are many advantages to using Global Shape Estimation. It has been shown that this is a very reliable and accurate method for determining the deflection of a static structure [1, 2, 3, 21]. Also, unlike methods methods such as curve fitting, it is very quick to implement requiring a simple matrix multiplication to get the results.

There are a number of reasons the dynamic mode shapes are used for the shape estimation. First of all, the dynamic mode shapes are the easiest to find. Finite element methods and dynamic models of the system find and use the dynamic modes. The dynamic modes also correspond to the lowest energy modes of the structure, which are also the preferred modes for deformation. Thus the dynamic modes are the modes that are determined by the structure itself.

2.4.2 Quasi-Static Shape Estimation

Quasi-Static Shape Estimation is the simplest way to create a shape estimation method for dynamic systems. The premise is to perform a separate static estimate of the structure's shape at each time step. The dynamics of the structure are not considered only the instantaneous readings of the sensors.

Although the dynamics of the structure are not considered, it is best to use the dynamic modes of the structure as the interpolation shapes. Once these modes are found the sensor readings for each mode is determined and put into a matrix, see equation 2.4. For a structure with n sensors, up to n modes can be sensed. The $\tilde{C}$ matrix[1] is inverted, or pseudo-inverse if fewer than n modes are being sensed, to produce the sensor matrix S. The sensor matrix is then used to determine the estimate of the modal amplitudes, $\hat{\eta}$, as shown in equation 2.6 . In the absence of noise or aliased modes this will produce exact results.

$$Z = \tilde{C}\eta \tag{2.4}$$

$$S = \tilde{C}^{-1} \tag{2.5}$$

$$\hat{\eta} = SZ \tag{2.6}$$

This method has a number of advantages over more complicated methods. The first advantage is that it is by far the quickest method and the least computationally expensive. If there are only a few positions that need a deflection estimate, such as tip deflection or pointing, then Quasi-Static estimates can even be programmed with analog devices.

Another advantage is that the Quasi-Static method does not require additional calculations to correctly sense a static offset in position. This is most useful for structures that have very low bandwidth loadings or for deformation loadings rather than force loadings. This method is also the least susceptible to changes in the structure, since most changes would change the dynamics of the structure but not

[1] $\tilde{C}$ is used because this matrix is a subset of the C matrix from the state space formulation.

the overall stress distribution.

The main disadvantage of Quasi-Static Shape estimation is that it is the most sensitive to aliasing of unsensed modes. If the bandwidth on the loading is below the frequency of the last sensed mode, then the quasi static method is usually sufficient. However as the noise bandwidth increases the effects of the aliasing increases as well.

2.4.3 Temporal Filtering for Quasi-Static Shape Estimation

The greatest disadvantage of the Quasi-Static method is its susceptibility to aliasing of higher modes, which will be shown in both the computer simulation chapter and the experimental chapter. The obvious solution to the aliasing is to filter those modes out. On a static structure the only method to do this is through shaped sensors, but on a dynamic structure it is possible through the use of a temporal filter. Algebraicly this is easy to show, by combining equation 2.6 with a filter F(t) as shown in equation 2.7.

$$\hat{\eta} = SZ * F(t) \tag{2.7}$$

Every dynamic deflection mode of a structure has a corresponding frequency, with the higher modes having higher frequencies. Using a low pass filter it is possible to filter out the high bandwidth signals from the sensors and allow the low frequencies to pass through. Setting the cutoff frequency of the filter to be between the sensed and the unsensed modes should reduce the effect the aliased modes have on the deflection shape and allow the sensed modes to go through unaffected. Incidentally the same result can be obtained by using the temporal filter after the shape has been estimated because the shape estimation method is linear.

The advantages of this method are considerable. The only difference between this method and the Quasi-Static method is the addition of a low pass filter. This means that the implementation is quite simple and the only additional information that is needed is the approximate modal frequencies.

The inherent disadvantage of this method is that information is lost. While it is desirable to ignore the higher modes to prevent aliasing, the information could still

be made useful in some other form.

An additional disadvantage is that any filter will add lag to the system. The amount of lag depends on the type of filter that is used. Generally the lag mainly effects the modes closest to the roll-off, so in systems with greater numbers of sensors and sensed modes, the effects will be less than for systems with few sensors.

2.4.4 Kalman Filtering

The final method that will be considered is the use of a Kalman filter to determine the modal amplitudes. The Kalman Filter [5] was designed for systems with noisy sensors where the plant dynamics are known *a priori*. The principle is to propagate the modal information forward for each time step. The sensor information readings are compared to the propagated states and new states are determined.

Design of a Kalman Filter starts with a state space model of the structure. Unlike the limited model used for Quasi-Static estimation, this model includes estimates of the errors and the forcing of the structure. For the estimation to be true least squares, the errors and forcing should be white noise with zero mean[2]. The states of the system are the modal amplitudes and the modal velocities. Unlike the Quasi-Static methods, the Kalman filter can estimate more modes than there are sensors, and it is advantageous to include as many modes as can be accurately modeled.

$$x_k = \Phi x_{k-1} + v \tag{2.8}$$

$$z_k = H x_k + n \tag{2.9}$$

The Kalman Filter uses the discrete state space model[3] as illustrated in equations 2.8 and 2.9. Eq's 2.10–2.14 show how the Kalman filter is used in practice. The (-) superscript indicates values that are propagated forward in time before new information is added, and the (+) superscript indicates the current values with the

[2]There are methods of designing Kalman filters for non-zero mean and colored noise, see Brown [5].

[3]Derivation of the variables from continuous state space form to discrete state space form is detailed in Appendix B or in Brown [6].

inclusion of the new information.

$$x_k^- = \Phi x_{k-1} \tag{2.10}$$

$$P_k^- = \Phi P_{k-1} \Phi^T + Q \tag{2.11}$$

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{2.12}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k (Z_k - H x^-) \tag{2.13}$$

$$P_k^+ = (I - K_k H) P_k^- \tag{2.14}$$

The H matrix is the sensing matrix, which contains the sensor readings if there is no sensor noise. The P matrix is the covariance of the uncertainties for the states. The most important value is K, which is also called the Kalman gain. The Kalman gain is the weight applied to the sensor readings, as opposed to using the previous estimate propagated forward in time.

There are a number of advantages to using a Kalman Filter. The first is that it has inherent anti-aliasing ability, in that the Kalman Filter is able to differentiate between signals of different frequencies even if they have the same spatial weightings. A Kalman filter does this by estimating more modes than there are sensors. Estimating additional modes also produces less errors due to truncation.

The Kalman Filter is also a good approach because it does not require filters or storage of previous data points. The amount of lag that is produced is also considerably less that for other methods of filtering.

Although in general the Kalman Filter is very good, it does have a few disadvantages. The first is that it requires a model of the system. If the model is inaccurate the Kalman Filters suffers more than other methods. The Kalman Filter also takes more time to run than the Quasi-Static system. Generally the Kalman Filter is still capable of running in real time, but for very high bandwidth systems alterations have to be made.

Inherent in the Kalman Filter is the assumption that the input noise and the structural forcing obey zero mean white noise restrictions. This is not generally the

case especially in loaded structures. Additional modes must be used to estimate the force bias [5].

2.5 Summary

There are a number of different shape estimation methods available, each with inherent advantages and disadvantages. More simple structures favor the more powerful methods, such as integration, where more assumptions about the behavior of the structure can be made safely. A more complicated structure favors simple methods such as global shape estimation.

Using a global shape estimation method, with the interpolation shapes equivalent to the mode shapes, allows the use of dynamic methods. The dynamic methods are designed, primarily, to alleviate the effects of aliasing. Methods including temporal filtering and Kalman Filtering are considered.

Chapter 3

Numerical Simulations

3.1 Introduction

In order to verify the usefulness of applying dynamic methods to shape estimation numerical trials were performed. The trials consisted of using a finite element model of a beam undergoing white noise excitation. The modeled beam includes models of strain sensors distributed along the beam.

The information from the model strain sensors is passed through the various dynamic methods to obtain estimates of the tip deflection. The results are compared to the average tip deflection.

There are three primary types of trials that were conducted. The first is a perfectly modeled structure with perfect sensors, using Different numbers of sensors. The second set of trials adds sensor noise of various levels. The final type of trial adds modeling error and sensor noise to try to better represent a real structure.

3.2 Beam Simulation

The canonical problem for shape estimation has always been the cantilevered beam. The beam is a simple enough structure that it is generally easy to model, and for a cantilevered beam the mode shapes and frequencies are well known. An additional advantage of using a cantilevered beam is that the tip deflection is an ideal metric to

check the accuracy of the method.

The beam modeled in the simulations corresponds very closely to the actual beam that was used in the experiments. The modeled beam was steel, measuring .381 meters in length, and 1.016 mm thick, with Young's Modulus 210 GPa, and specific weight of 7.850. The resulting fundamental frequency is 36.7 rad/sec. The forcing used in the simulations was a transverse tip force. This does not mirror the experiment, described in the next chapter, because the tip loading provides clearer transfer functions for analysis as well as ensuring that all modes are forced.

The simulations performed here use a finite element model of a cantilevered beam with 240 Bernoulli Euler Beam finite elements. The number of elements was set to allow ample resolution and accurate mode shapes. The first 25 modes of the structure are retained for the dynamic simulation of the structure. Fewer modes would have made the modeling inaccurate due to truncation errors, and more modes were determined to be superfluous since the contribution to the deflections and strains were so small.

Each of the cases was run for a system with 1, 2, 4, 6, 8, and 10 sensors that are evenly distributed along the length of the beam with the first sensor always going at the root, i.e. for 4 sensors they go at the root, at 1/4 L, 1/2 L and 3/4 L. These locations are shown in Figure 3-1 overlaid on the second displacement mode. The sensors are modeled as being 4 elements in length or 1/4 inches in length. The sensor reading is equal to the difference in slopes at the edges of the elements, which corresponds to the integral of the curvature.

All of the computer simulations for the beam were performed in Matlab and the source codes have been included in Appendix C. The first step is to calculate the mode shapes, natural frequencies and sensor weightings based on the finite element model. This information was used to determine the state space matrices, A, B, and C, with the D matrix being zero. The model was then run through a state space simulator to obtain the modal amplitudes and sensor readings as a function of time for a given random disturbance. The results of these simulations were used in each shape estimation method and compared.

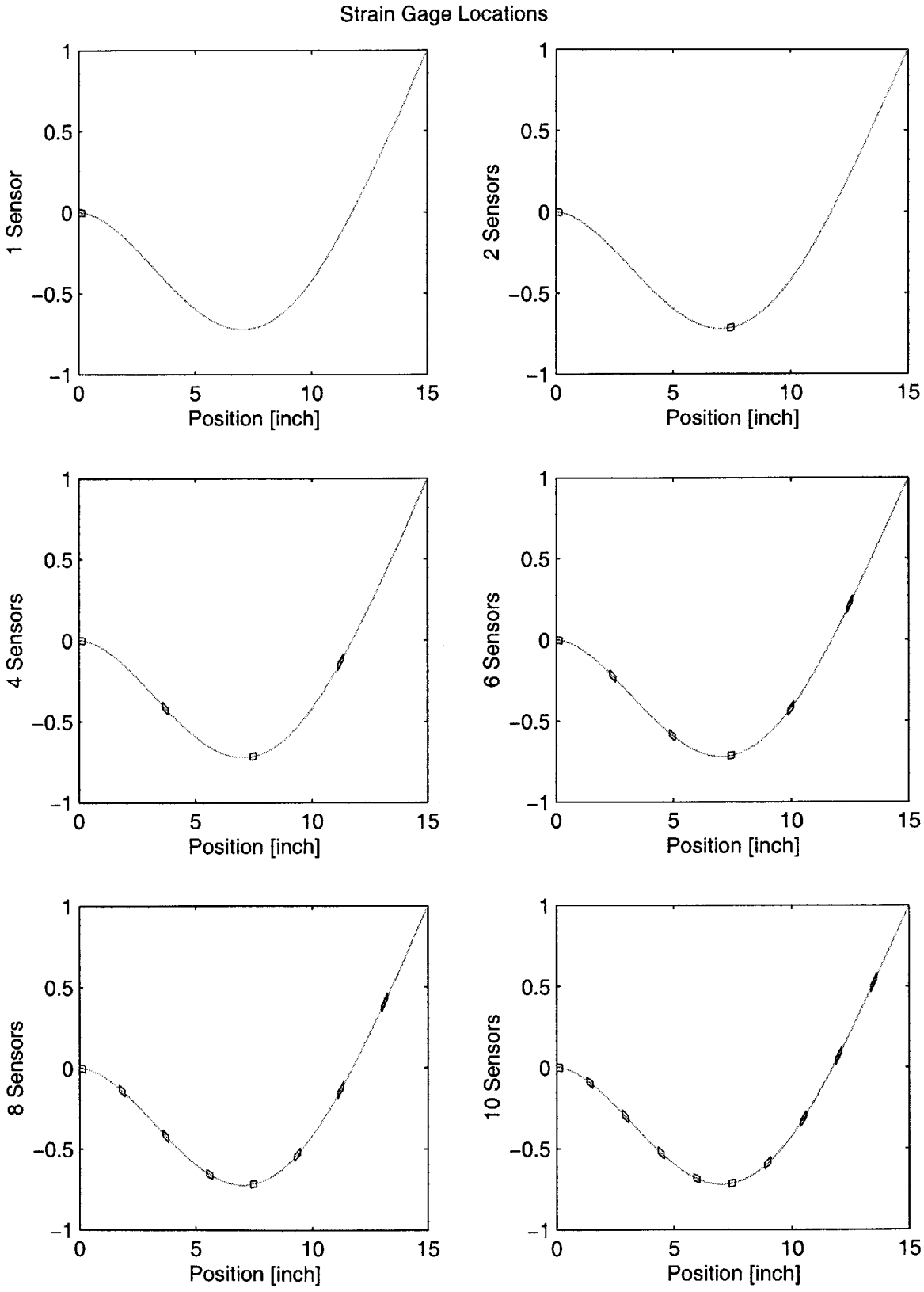


Figure 3-1: Sensor Locations for 1, 2, 4, 6, 8 and 10 Sensor Cases

To avoid the effects of startup transients, and to more accurately model the continuous estimation of a structure, the first half of the simulated results were not used. The shape estimation was then performed on the remaining 16,384 (2^{14}) time steps, for a complete simulated runtime of one second.

The time step was chosen to allow sensing of the full twenty-five included modes. A shorter time step would include unmodeled frequencies which would not effectively change the dynamics of the problem. The total run time was chosen to allow multiple oscillations of the fundamental mode. The results of twenty-five separate simulation runs are averaged to determine the results.

The errors are calculated by finding the RMS difference between the actual tip deflection and the estimated tip deflection. To find a percent error, the RMS error is normalized by the RMS tip deflection. The tip deflection is the chosen metric both to correspond with the experiment, and because the tip of the beam is the point of greatest deflection for each mode. Additionally, when using mass normalized deflection shapes the tip deflection is identical for all the modes.

After running each of the methods on perfectly modeled system with no induced errors, a number of additional simulations were performed. The first is an investigation into the effects of sensor noise. The noise levels used are 1%, 5%, 10%, and 20%, where the noise is a percentage of the RMS reading for the root strain gage. The second set of tests were run on a system that includes up to 5% modeling error. The level of modeling error was chosen to correspond to expected levels for a realistic structure.

3.3 Quasi-Static Shape Estimation

The Quasi-Static shape estimation method is the simplest method. The necessary weights were calculated by inversing the first portion of the C matrix used in the state space model. The resulting matrix $S_{n\times n}$, where n is the number of sensors, is multiplied by the sensor readings to obtain the estimate of the first n modal amplitudes. These amplitudes are then multiplied by the deflection shapes to obtain the estimate

of the beam's shape. This process is repeated for each time step of the simulation.

The model was run for a number of different cases. The first case was for a perfectly modeled structure with no sensor errors, while not a case that will ever be experienced in a laboratory it will provide a baseline how the other errors enter the system. The second case is for a perfectly modeled structure with various levels of sensor errors. The final case is for an incorrectly modeled structure with and without various levels of sensor errors.

3.3.1 Ideal Model with no Sensor Noise

For the ideal case of Quasi-Static shape estimation, the model behaved as predicted. There was a very large level of error when the sensor count is low and it drops off with increases in the the number of sensors. Using only a single sensor the deflection error is greater than 100%. It takes eight sensors before the error drops to 10%!

The Quasi-Static estimation errors initially drops off proportional to $1/n$, where n is the number of sensors. For higher number of sensors the error drops off proportionally with $1/n^2$. This is most likely an effect of model truncation and digitization of the higher frequencies. The results of these effects is to lower or eliminate the response to the higher modes, reducing their ability to cause errors by aliasing.

The aliasing effect is seen even more clearly in the case where there is only one sensor, as illustrated in Figure 3-2, which shows the estimated displacement and the actual displacement. The estimate rolls off at $1/\omega$, while the actual displacement rolls off at $1/\omega.^2$. The result of this is that the high frequency modes have tremendous influence on the estimate.

For the Quasi-Static case with no noise, a fully analytic model of the structure is possible. Thus the results of the simulation can be compared to that of the predicted Bode plots for the system as shown in Figure 3-2. In the predicted plots, notice how the actual displacement rolls of as $1/\omega^2$ while the static estimate rolls off as $1/\omega$ over the entire bandwidth.

The results are shown in Table 3.3.2, which includes the data with sensor noise. The cause of the errors becomes obvious when the Fourier Transform of the resulting

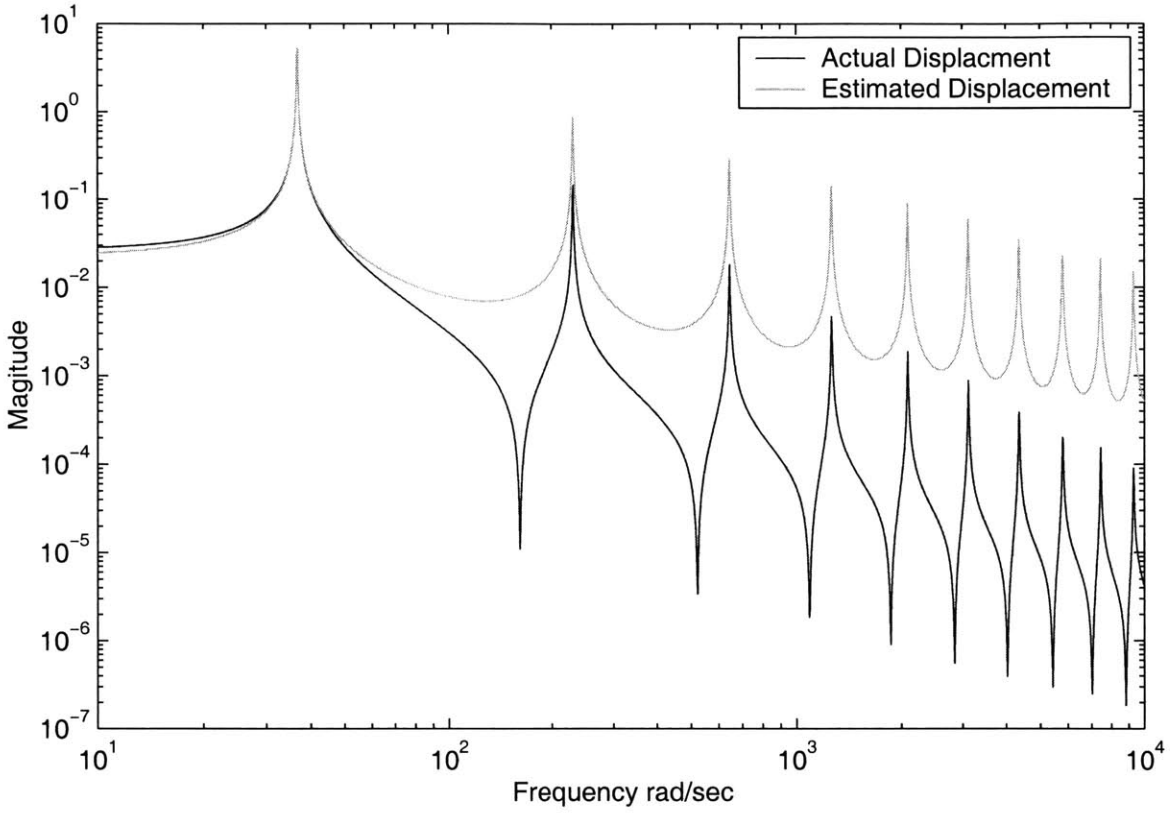


Figure 3-2: Bode Magnitude Plot of Actual Deflection vs. Estimated Deflection For Quasi-Static Shape Method with 1 Sensor

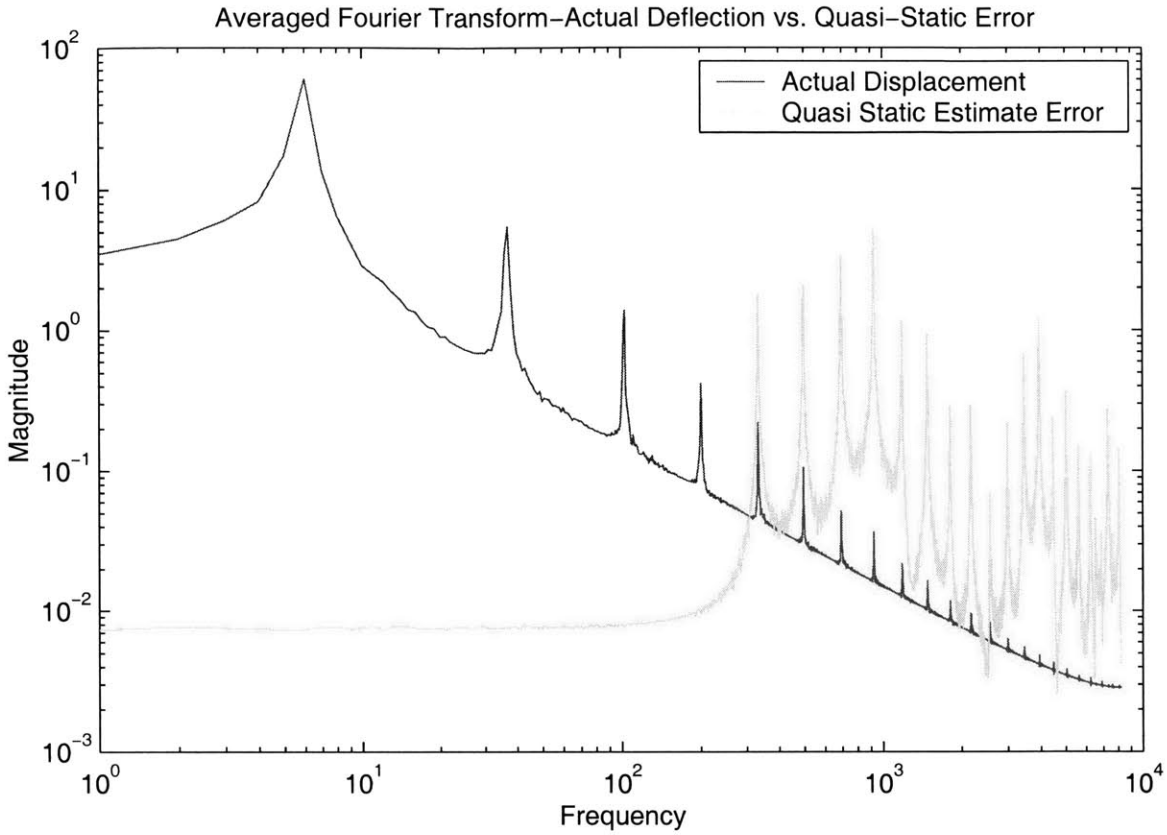


Figure 3-3: Average Fourier Transform of Actual Deflection vs. Estimate Error for Quasi-Static Shape Method with 4 Sensors

deflections are examined. Figure 3-3 shows the averaged, over 25 trials, Fourier transform for the actual displacement and the displacement error for the four sensor case. The low frequency error is two and a half orders of magnitude less than the deflection, but at high frequencies the error is nearly that many times greater.

To get a clearer picture on the effects of the aliasing it is helpful to look at a time history of the predicted and actual deflection. Figure 3-4 shows such a case. The top graph shows the complete time history of the predicted deflection and the estimated deflection, again for the four sensor case. What appears to be noise at that level is clearly seen to be higher order harmonics when zoomed in on the region indicated by the box, as shown in the bottom half of the figure.

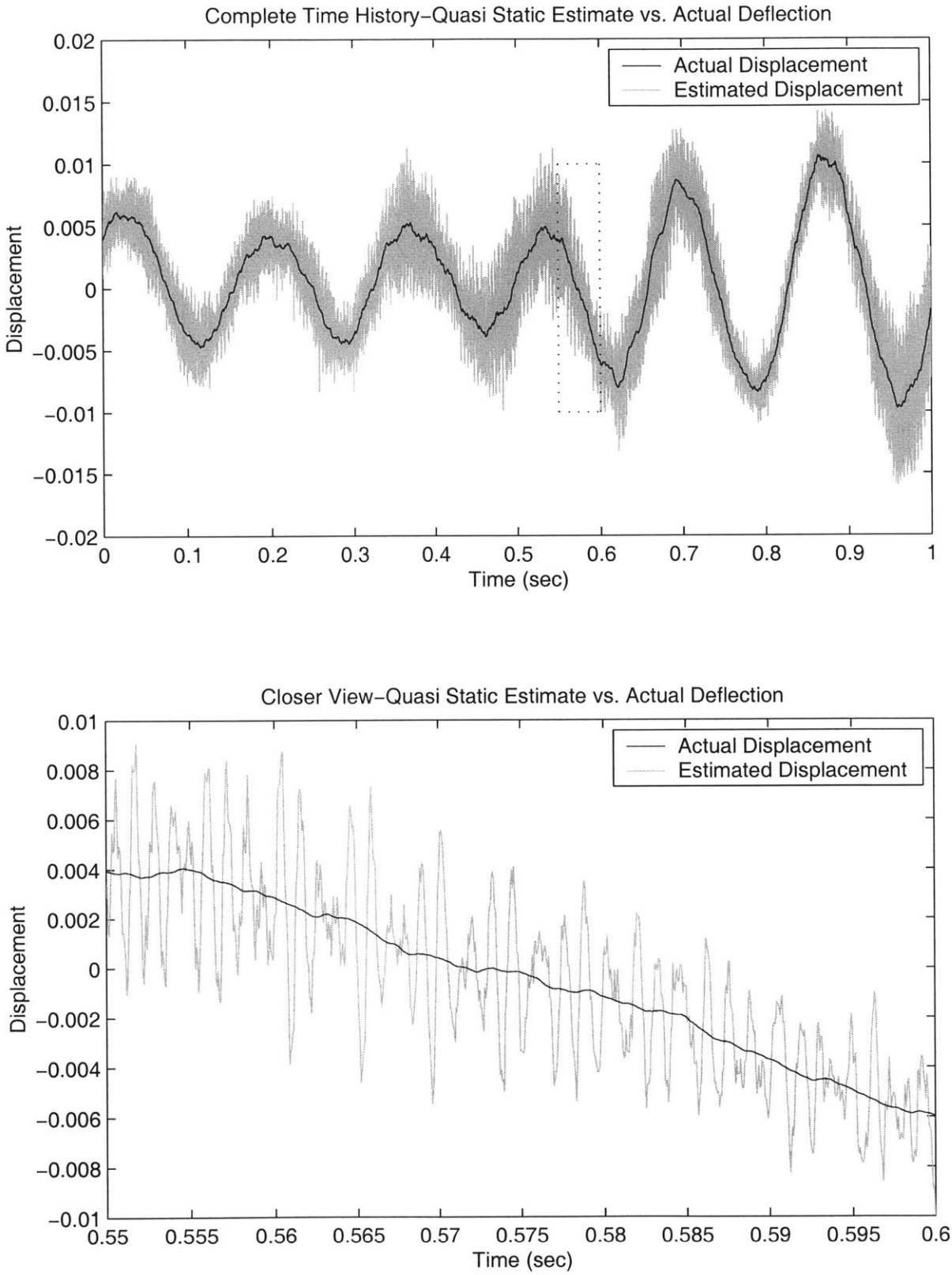


Figure 3-4: Time History of Actual Deflection and Quasi-Static Estimate

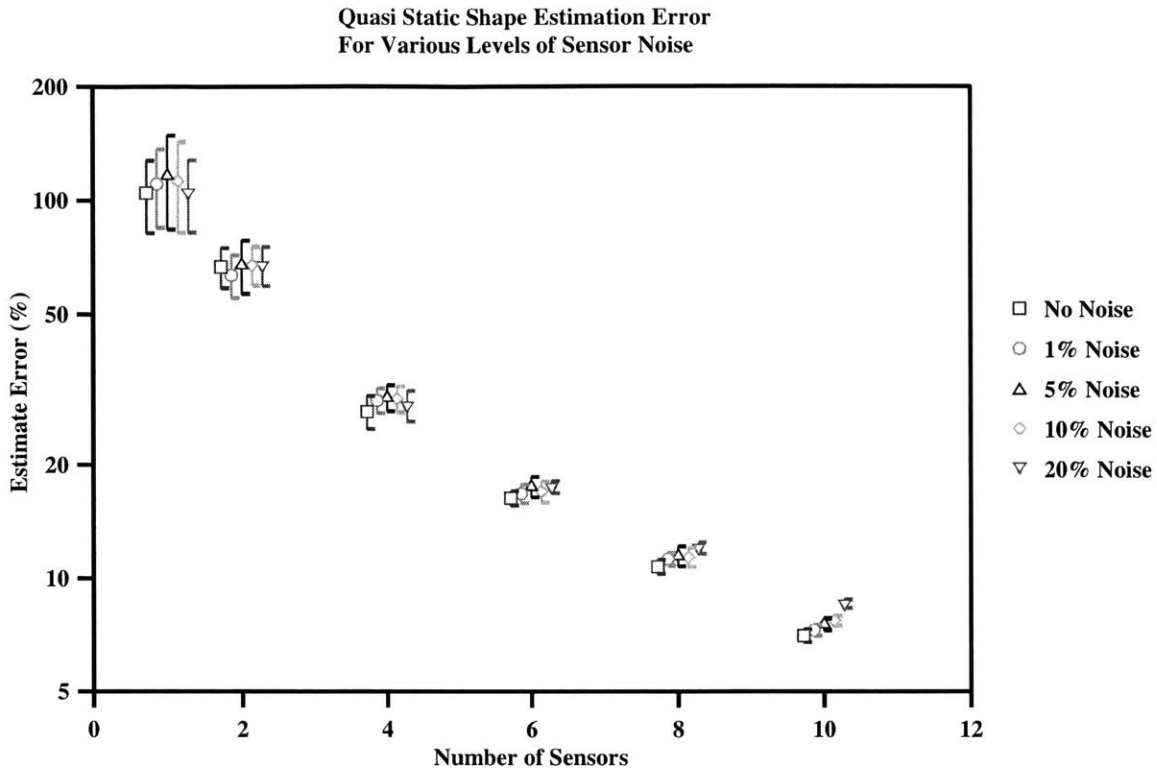


Figure 3-5: Estimation Error as a function of Sensor Noise and Number of Sensor for Quasi-Static Shape Method

3.3.2 Ideal Model with Sensor Noise

The next step is to add sensor noise to the simulation. The sensor noise was added as a percentage of the of the RMS sensor reading for the root strain gage. The sensor reading for the root strain gage was chosen because it is used in all of the sensor layouts.

The effects of the sensor noise on the Quasi-Static estimate is actually very small. For the cases where there are 1, 2, and 4 sensors the difference between the estimate errors is actually in the noise floor of the simulation. Only for the 10 sensor case is there a statistically clear distinction between errors. In that case the difference is between 7% error for the case without noise and 8.5% for the case where the sensor noise is 20%. The percent errors are shown in Table 3.3.2, and Figure 3-5 shows the estimate error versus the number of sensors for the various levels of noise.

The reason that the sensor errors do not have a significant effect except in the

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	104.74	67.02	27.69	16.34	10.74	7.02
1 %	110.65	63.70	29.63	16.80	11.26	7.26
5 %	116.22	67.82	30.12	17.52	11.47	7.53
10 %	112.50	67.92	29.92	17.03	11.39	7.71
20 %	105.07	67.79	28.71	17.50	12.06	8.57

Table 3.1: Quasi-Static Estimation Error due to Sensor Noise

cases with a high number of sensors becomes clearer when examining the averaged Fourier transform of the system as shown in Figure 3-6 for the 4 sensor case with 20% error. The low frequency portion of the graph is virtually unchanged from previous, while for the high frequency the noise floor masks the signal. When using a limited number of sensors the effects of aliasing cause a similar disturbance level of error as the sensor noise. For a greater number of sensors, the model tries to predict modes that are partially below the noise floor which causes the addition of errors.

3.3.3 Effects of Modeling Errors

While sensor noise is a large cause of error, it is not the only cause. Inherent in any physical system are structural defects, which a model of the structure does not include. These defects range from statistical variations in the modulus, thickness, and density of the components to misplacement and alignment of sensors, and structure damage which can vary over the lifetime of the structure.

In an attempt to model possible errors to the structure the sensing matrix was randomly changed by a small percentage of the mean sensor reading for that mode. For some entries this means a very small net change, while for sensors that are near a node for a particularly mode, it can cause a sign reversal.

The simulations were performed with 5% modeling error and 20% modeling error. While neither error is that significant, they are on the right order for most applications. Figure 3-7 shows the effects that 5% and 20% modeling error has on the estimate error. The figure includes the correctly modeled system for comparison.

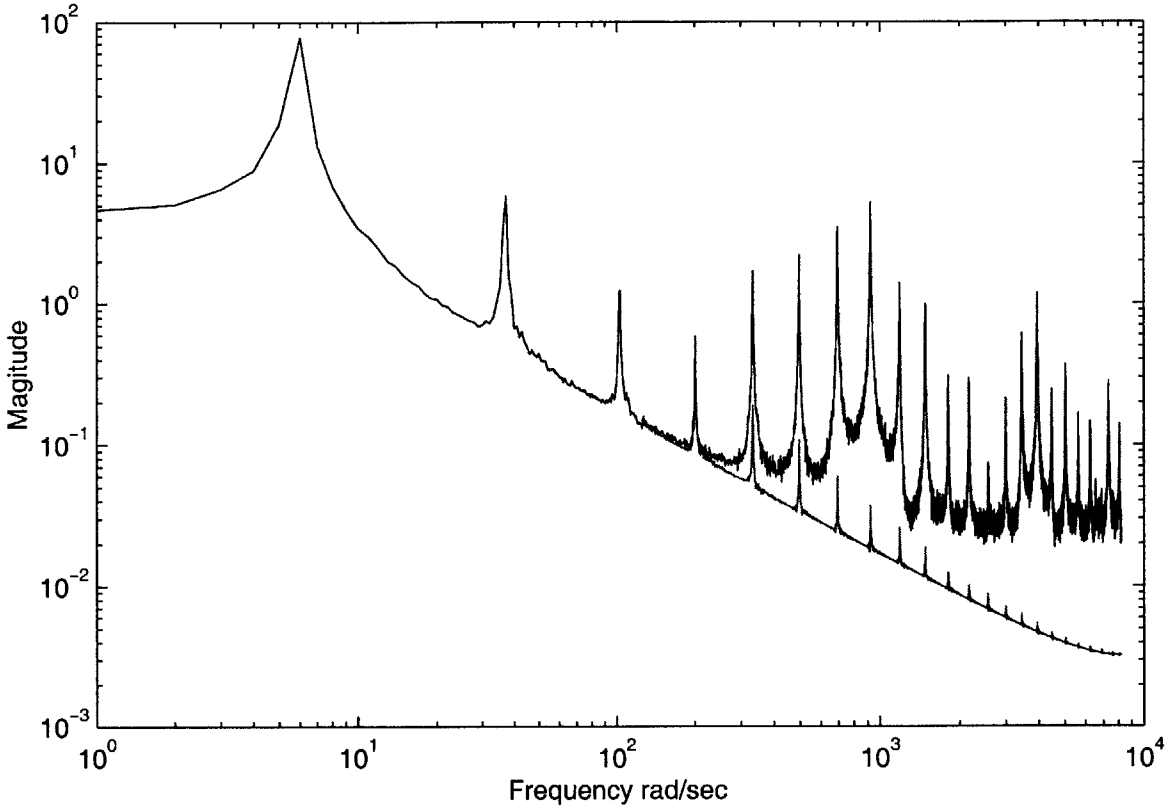


Figure 3-6: Fourier Transform of Quasi-Static Estimate with 20% sensor noise and 4 sensors

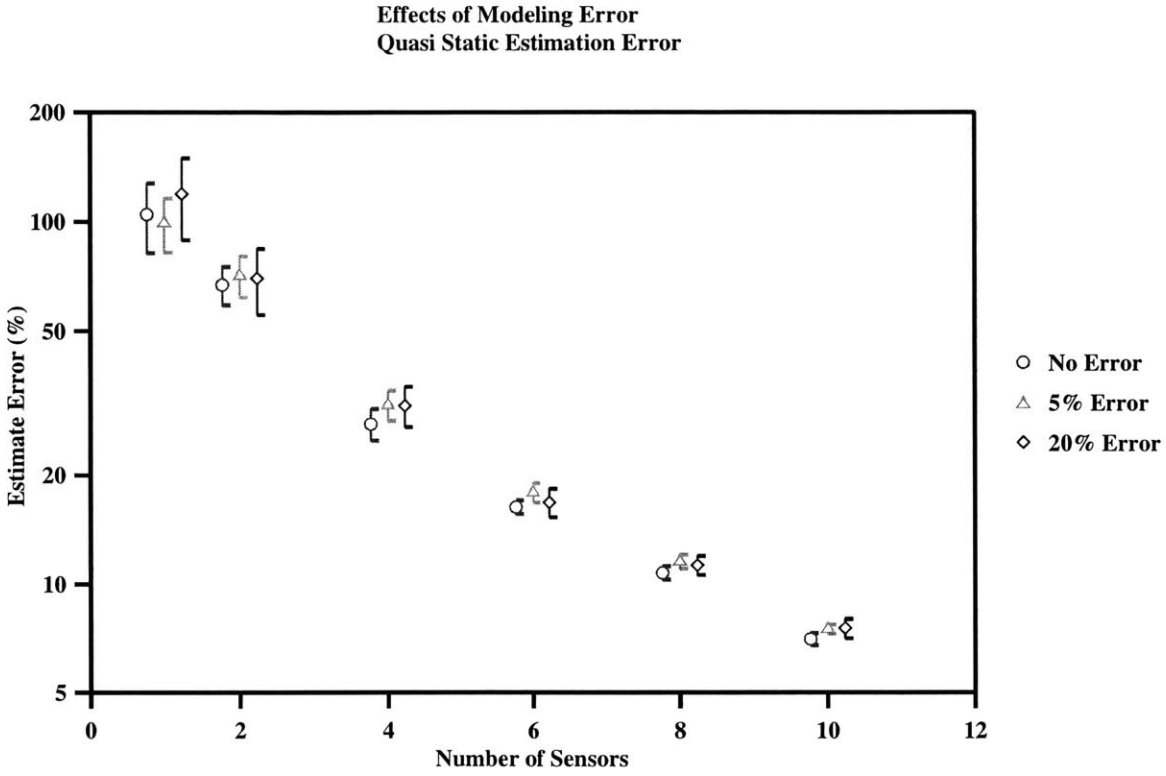


Figure 3-7: Results of Modeling Error on Quasi-Static Shape Estimation Error

The additional of modeling error actually has very little effect on the error resulting from the Quasi-Static method, see Table 3.3.3. The only significant error occurs for the case where there is no sensor noise. Even with low or no sensor noise the errors from modeling only have an effect when the errors from aliasing have been lessened by having multiple sensors.

Sensor	Number of Sensors					
Noise	1	2	4	6	8	10
0 %	104.74	67.02	27.69	16.34	10.74	7.02
5 %	99.16	71.07	31.25	17.94	11.55	7.48
20 %	119.27	69.85	31.12	16.84	11.28	7.53

Table 3.2: Quasi-Static Method, Effects of Model Error

3.4 Quasi-Static Shape Estimate with Temporal Filter

In an attempt to alleviate the effects of aliasing of high frequencies a temporal filter was added. The filter is a low pass filter whose roll-off frequency is set between the last mode sensed and the first aliased mode. In the simulations the sensor data was filtered, but the same effect can be gained by filtering the output signal.

Unlike the unfiltered Quasi-Static method, the method with filtering has transients that occur at the end points of the data. If included these transients can cause tremendous increase in the level of the error. Since the transients are only a relic of the simulation method the end points were thrown out and the RMS errors were calculated from the central data points only.

The same trials were used for filtered case as for the Quasi-Static case. This allows direct comparison of the two methods and reduces the effects of having a limited sample space.

3.4.1 Ideal Model with no Sensor Errors

For the filtered Quasi-Static case with no modeling errors, or sensor noise the system behaved considerably better than without filtering. For this case the greatest source of errors was due to truncation of the system. While the higher frequency modes contribute significantly less to the overall deflection, they do still contribute.

Figure 3-8, shows the percent error in the estimation as a function of the number of sensors. The error is proportional to $1/n^2$. There is a slight deviation for the eight sensor case, however. This is most likely because the sensor locations are less advantageously placed for the eight sensor case.

3.4.2 Ideal Model with Sensor Errors

An interesting phenomenon occurs when sensor errors are added to the filtered Quasi-Static estimation. The performance actually decreases with the addition of extra

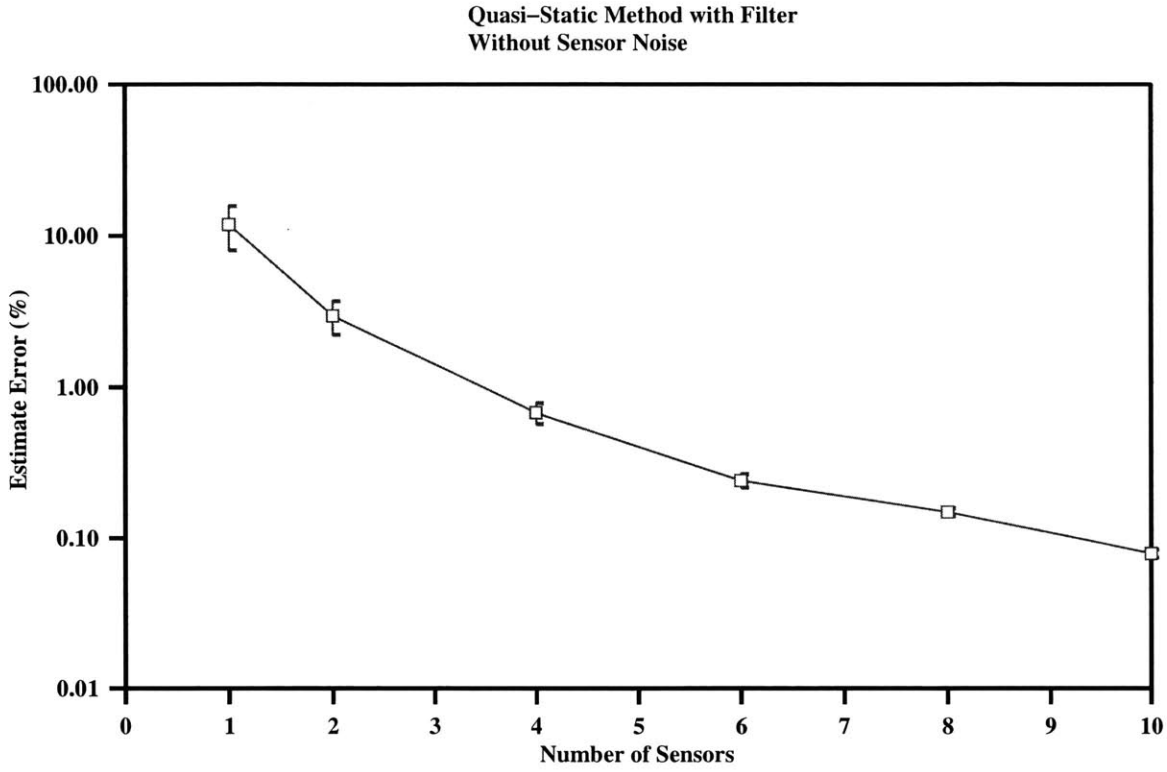


Figure 3-8: Tip Deflection Estimate Errors for the Filtered Quasi-Static Method

sensors.

Figure 3-9, shows the estimate errors as a function of number of sensors for various levels of noise. For low numbers of sensors the sensor noise has no noticeable effect on the total estimation error. As the number of sensors increases, it becomes apparent where the crossover from truncation error to sensor noise error occurs.

While this seems counter-intuitive it actually is a very reasonable result. Since the cutoff frequency is determined by the last mode that is sensed, the frequency increases with the addition of sensors. As the frequency increases more of the noise is included in the sensed bandwidth. The resulting errors are still considerably less than what is obtained using the static method without the benefit of a filter.

Although for 20% sensor noise, the error increases from 0.8% to 2.2% error for the ten sensor case. While this is a large increase it is still small compared to the results without filter of over 7% even without sensor noise. Table 3.4.2 lists the resulting errors for the various sensors and levels of sensor noise.

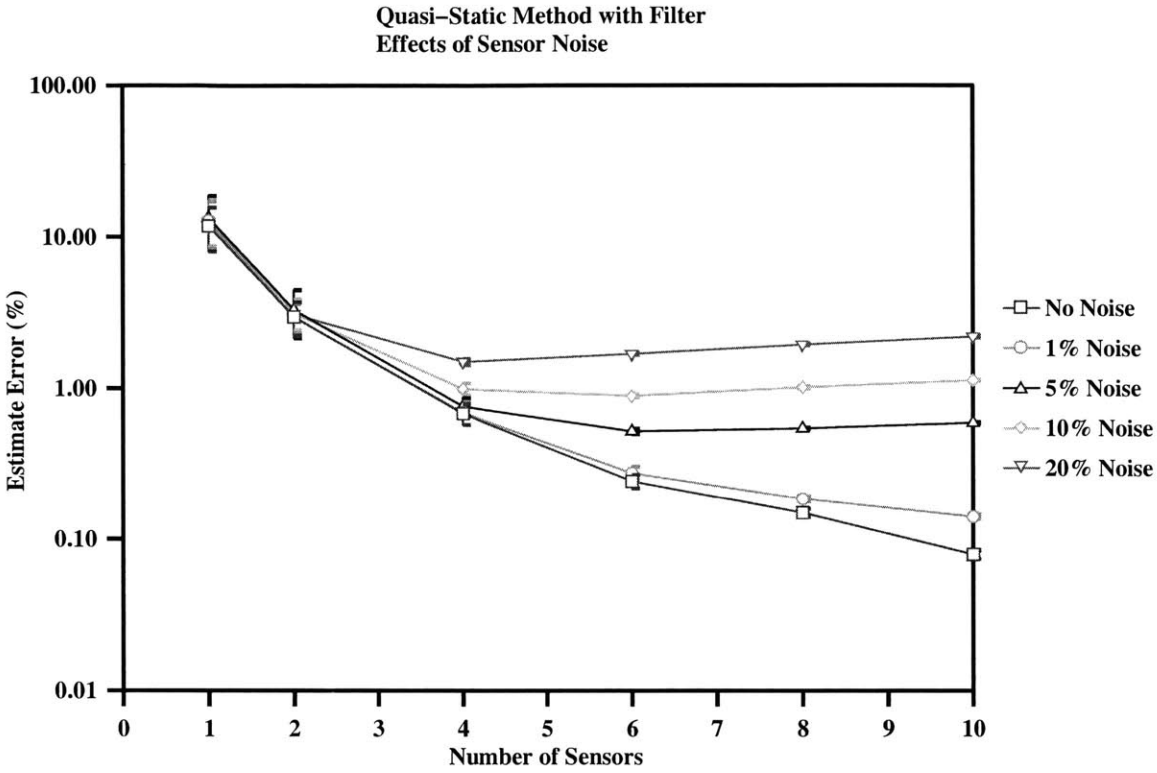


Figure 3-9: Sensor Noise effects on Quasi-Static Estimate with Filtering

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	11.88	2.97	0.68	0.24	0.15	0.08
1 %	12.80	2.97	0.69	0.27	0.18	0.14
5 %	13.43	3.31	0.76	0.52	0.54	0.59
10 %	13.35	3.14	0.99	0.89	1.02	1.13
20 %	11.88	3.10	1.50	1.70	1.96	2.20

Table 3.3: Filtered Quasi-Static Estimation Error, Effects of Sensor Noise

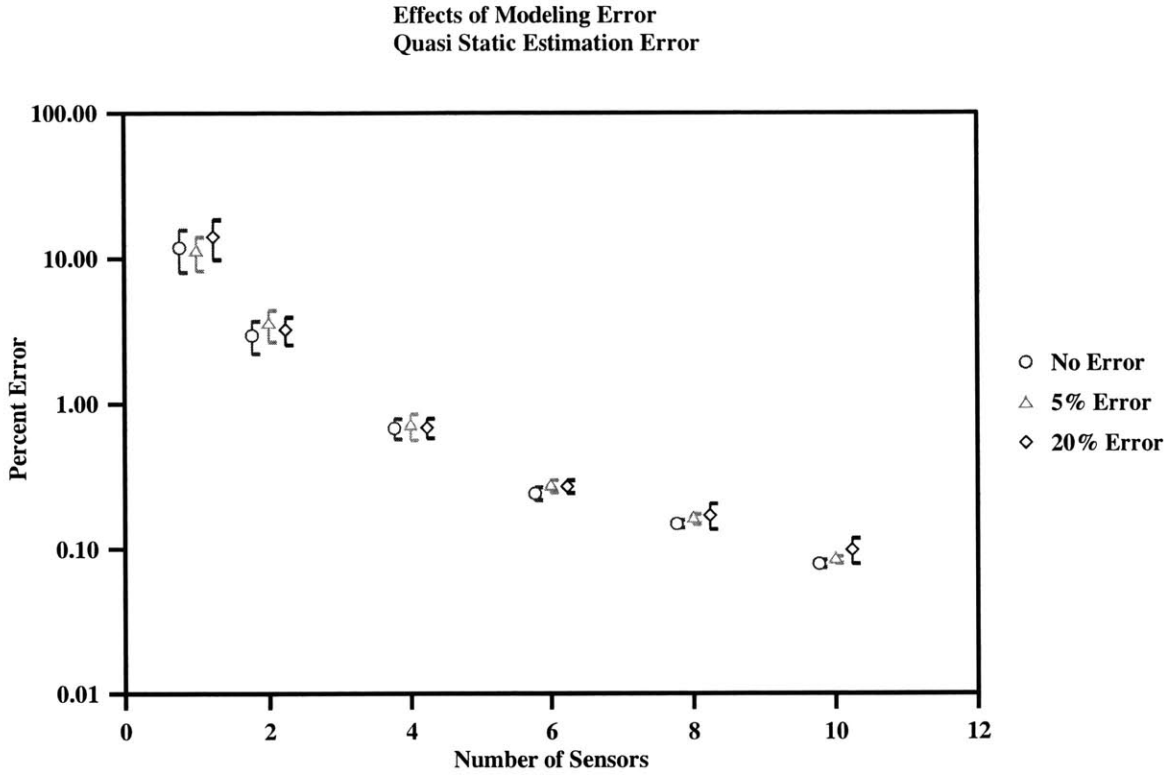


Figure 3-10: Effects of Modeling Error on Quasi-Static Estimate with Filtering

3.4.3 Effects of Modeling Errors

The same procedure was used to find the effects of modeling errors for the filtered case as for the unfiltered case. The results were also much the same. Small changes in the physical plant seems to have almost no effect on the resulting deflection estimation errors.

Figure 3-10, shows the error resulting from modeling errors for the filtered case. There is no consistent meaningful change in the resulting errors. See Table 3.4.3, for a comparison of the actual values.

3.5 Kalman Filter

The Kalman filter is a much different way to solve the problem, requiring more of the system, and more intensive calculations. Also unlike the other methods, the Kalman filter can actually sense more modes than there are sensors. To reflect this ability of

Model Error	Number of Sensors					
	1	2	4	6	8	10
0 %	11.88	2.97	0.68	0.24	0.15	0.079
5 %	11.13	3.54	0.71	0.27	0.16	0.084
20 %	14.13	3.26	0.69	0.27	0.17	0.099

Table 3.4: Filtered Quasi-Static Estimate Error, effects of Modeling Error

the Kalman filter, the simulations were run twice, once for a Kalman filter that only includes the model for as many modes as there are sensors, the Terse Kalman Filter, and a second run that has a greater number of modes, the Full Kalman Filter.

The expanded version of the Kalman Filter includes up to twice as many modes as the limited version, with the minimum modes included being five. In all cases such a dramatic increase in the number of modes might not be possible, since the time to iterate through one loop of the Kalman filter scales with the cube of the number of modes that are carried since the Kalman filter requires matrix inversion.

Figure 3-11 shows how advantageous having extra modes can be. The top plot shows the complete time history of the beam deflection, with the two Kalman estimates. The bottom plot is a closer look at the boxed region from the top plot. The beam was modeled with one sensor, so the Terse Kalman Filter has only one mode modeled. The Expanded Kalman Filter uses five modes, and is clearly much more accurate.

The Kalman filter has two variables that are not present in the previous methods that are essential to its working properly. The first is the estimate of the forcing on the structure. The Kalman filter uses the estimated level of the forcing to determine how much the propagated signal is likely to change between time steps. In the simulation this is known *a priori*, whereas for a real system it would have to be determined.

The second variable is the estimated level of the sensor error. In this case the sensor error level that is used in the Kalman filter needs to include the effects of the errors from the aliasing as well as any sensor noise. The final levels used were found by trial and error using the system without noise or modeling errors. The actual

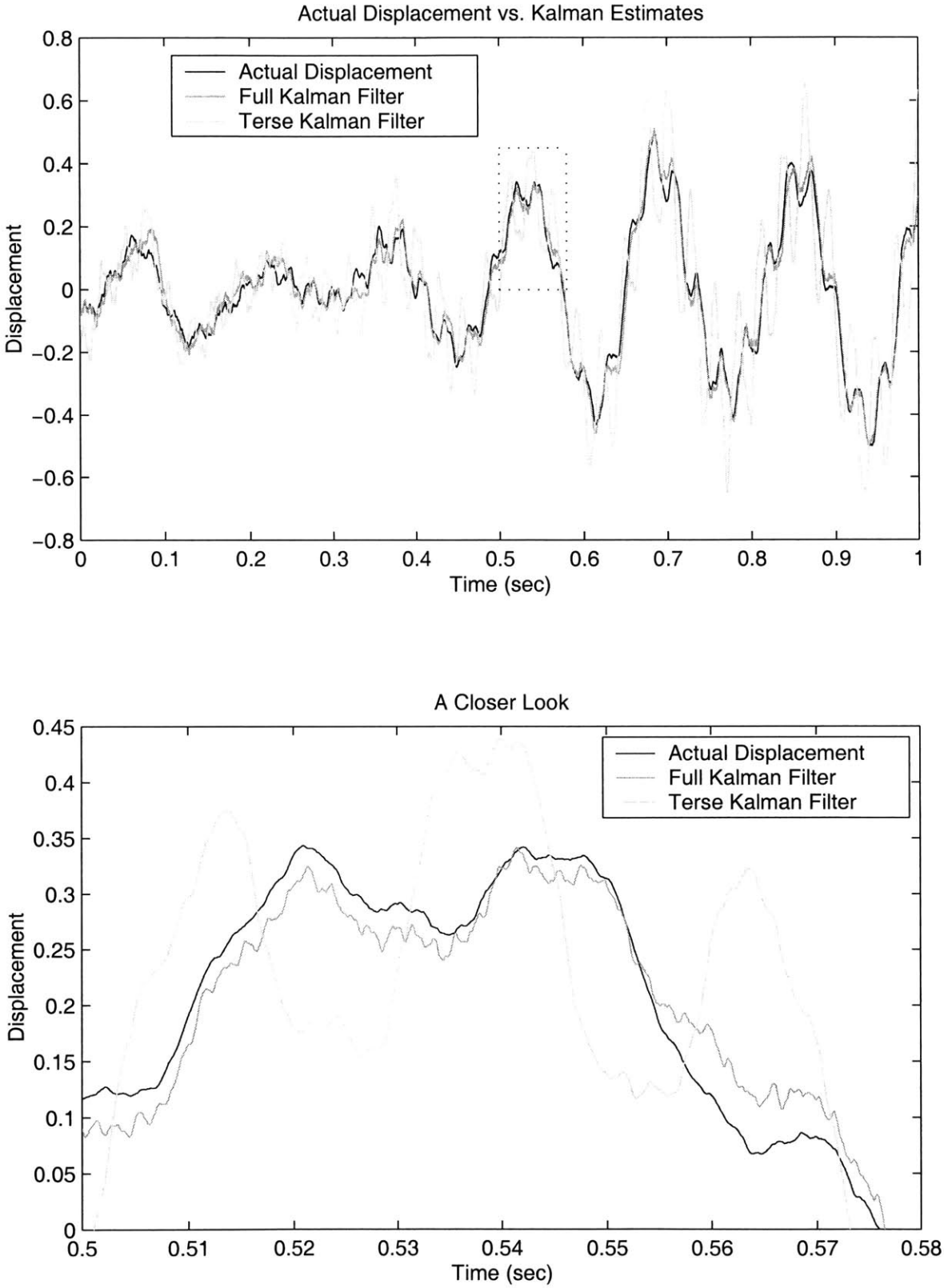


Figure 3-11: Time History of Actual Displacement and Kalman Estimates

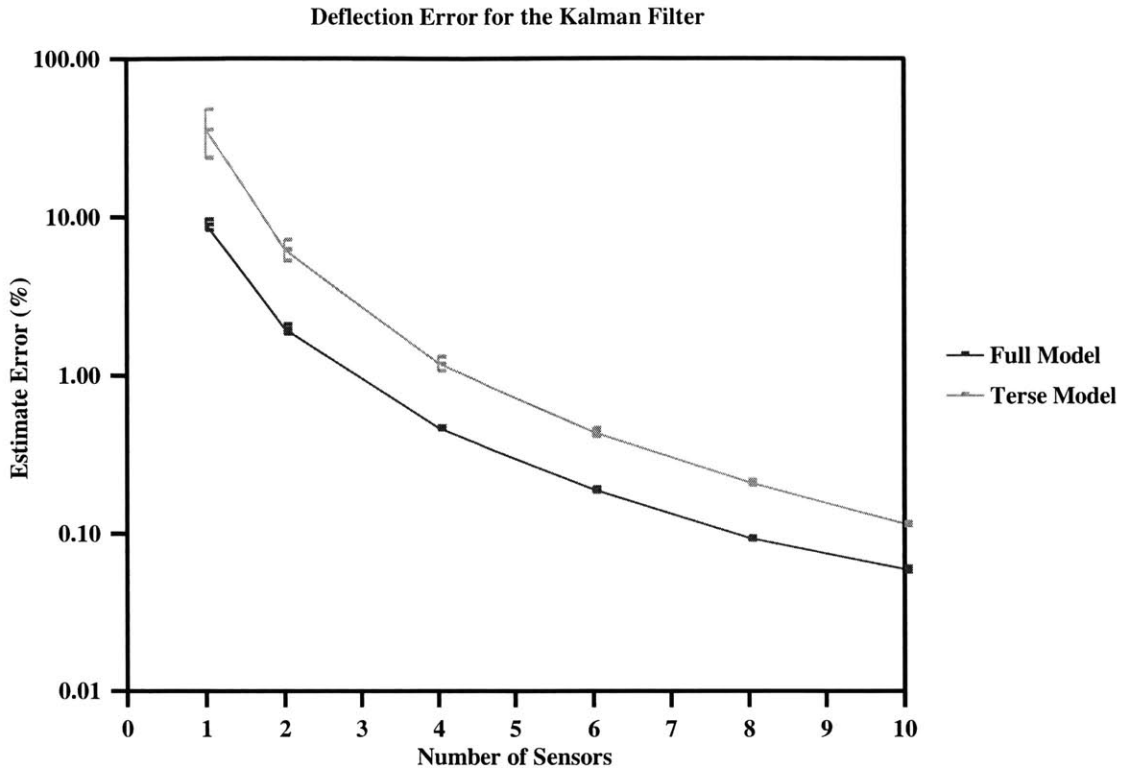


Figure 3-12: Kalman Filter Error for System without Sensor Noise

sensor noise ends up contributing only about 2% for the most extreme case.

Like the filtered Quasi-Static method, the Kalman filter has transient effects. The Kalman filter requires initial values for the estimated position and for the estimated error covariance. To account for the transients in the covariance matrix, the filter was run for a number of iterations before applying it to the data. This has the effect of making the filter seem like it has been running a long time. The initial values of the modal amplitudes used for the Kalman filter were the actual modal amplitudes. So as not to give the Kalman filter an unfair advantage the initial data points were not used in determination of the errors.

3.5.1 Ideal Model with no Sensor Noise

The Kalman filter did a very good job of reducing the effects of aliasing. Figure 3-12, shows the plot of the errors versus the number of sensors for the two versions of the Kalman filter.

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	36.0	6.36	1.20	0.44	0.21	0.12
1 %	39.2	6.07	1.23	0.46	0.22	0.12
5 %	40.9	6.60	1.25	0.50	0.26	0.17
10 %	40.3	6.45	1.30	0.54	0.34	0.26
20 %	36.0	6.40	1.36	0.72	0.54	0.47

Table 3.5: Terse Kalman Filter, Effects of Sensor Noise

Like the static filter the errors for both versions of the Kalman filter are proportional to $1/n^2$. The number of sensors is proportional to the modes sensed. As the number of sensed modes increases, the percent of the displacement that is not correctly accounted for drops off as $1/\omega^2$. Thus the error for the Kalman filter is proportional the maximum frequency that is correctly sensed.

The Kalman Filter that utilizes twice as many modes has errors approximately one third that of the Kalman Filter with limited modes. The cause of the additional errors is aliasing of the initial modes that are not modeled. Figure 3-13 shows the Fourier transform of the estimation errors for the four sensor case. The top plot shows the Kalman Filter error when the model includes the four modes, the bottom plot shows the effect of having eight modes in the model. Notice how the errors dramatically increase after the last mode of the model.

3.5.2 Ideal Model with Sensor Noise

The addition of sensor noise decreases the effectiveness of a Kalman filter, as is expected. For the 1 and 2 sensor cases there is very little effect, for the same reasons as for the Quasi-Static case. Unlike the Filtered Quasi-Static, case the effect of sensor noise effects comes more gradually. Table 3.5.2 shows the effect of sensor noise on the Kalman filter with the same number of modeled modes as sensors, the Terse Kalman Filter. Table 3.5.2 shows the results for the Full Kalman Filter which includes extra modes.

As the number of sensors increases, the effect of the noise becomes gradually more

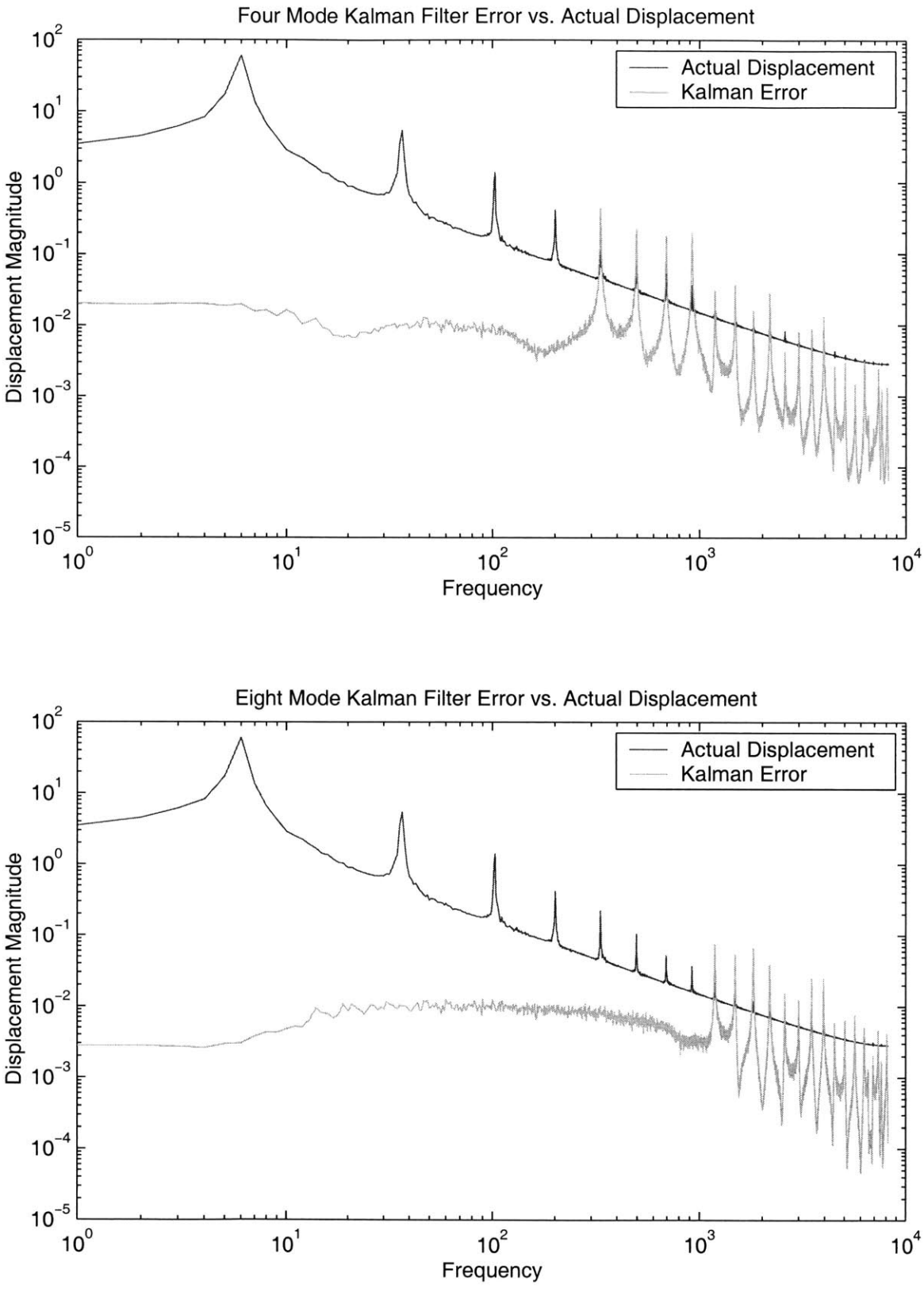


Figure 3-13: Comparison of Fourier Transforms for 4 and 8 mode Kalman Filter

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	9.12	2.01	0.47	0.19	0.094	0.059
1 %	9.14	2.10	0.49	0.20	0.10	0.069
5 %	9.44	2.18	0.53	0.25	0.17	0.14
10 %	8.87	2.12	0.60	0.35	0.28	0.24
20 %	9.14	2.16	0.82	0.61	0.52	0.47

Table 3.6: Full Kalman Filter, Effects of Sensor Noise

apparent. For high levels of noise the two versions of the Kalman filter asymptote to the same level of error. The sensor readings from the extra modes become completely lost in the noise. The result is that the extra modes become superfluous. This is clearly seen in Figure 3-14, which shows the results for the 20% sensor error case. For ten sensors the results are nearly identical. The plot is on log-log scale to emphasize the $1/n^2$ relationship between sensors and error. It also more clearly shows that the errors for both cases seem to asymptote to a constant value in the presence of noise.

3.5.3 Effects of Model Error

The Kalman Filter was less susceptible to sensor noise than the other methods, but it proves to be more susceptible to modeling error. This is because the Kalman filter relies more heavily on the model of the structure to predict its behavior. This is especially true of the Terse Kalman Filter which takes a severe loss especially for the 1 sensor case as shown in Table 3.5.3

Figure 3-15, shows the results for the case where the Kalman filter has the exact number of modes as sensors, the Terse Kalman filter. The plot is log-log to show that the level of errors is still directly dependent on the number of sensors. With the introduction of 5% error in the model the error increases dramatically for low sensors, but not as much for the higher number of sensors. The addition of 20% model error has almost identical results to the 5% model error case. The error is approximately proportional to $1/n^{2.75}$. See Table 3.5.3 for the numerical results for the Full Kalman Filter.

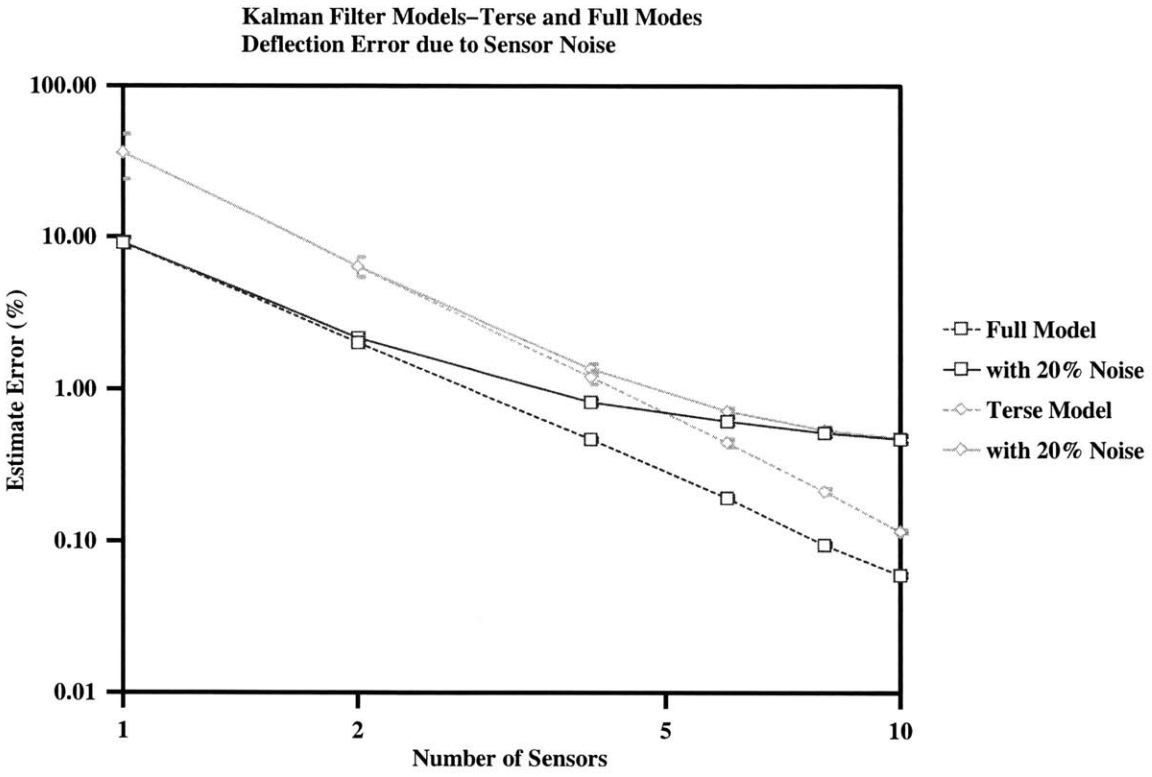


Figure 3-14: Kalman Filter Errors without Noise and with 20% Noise

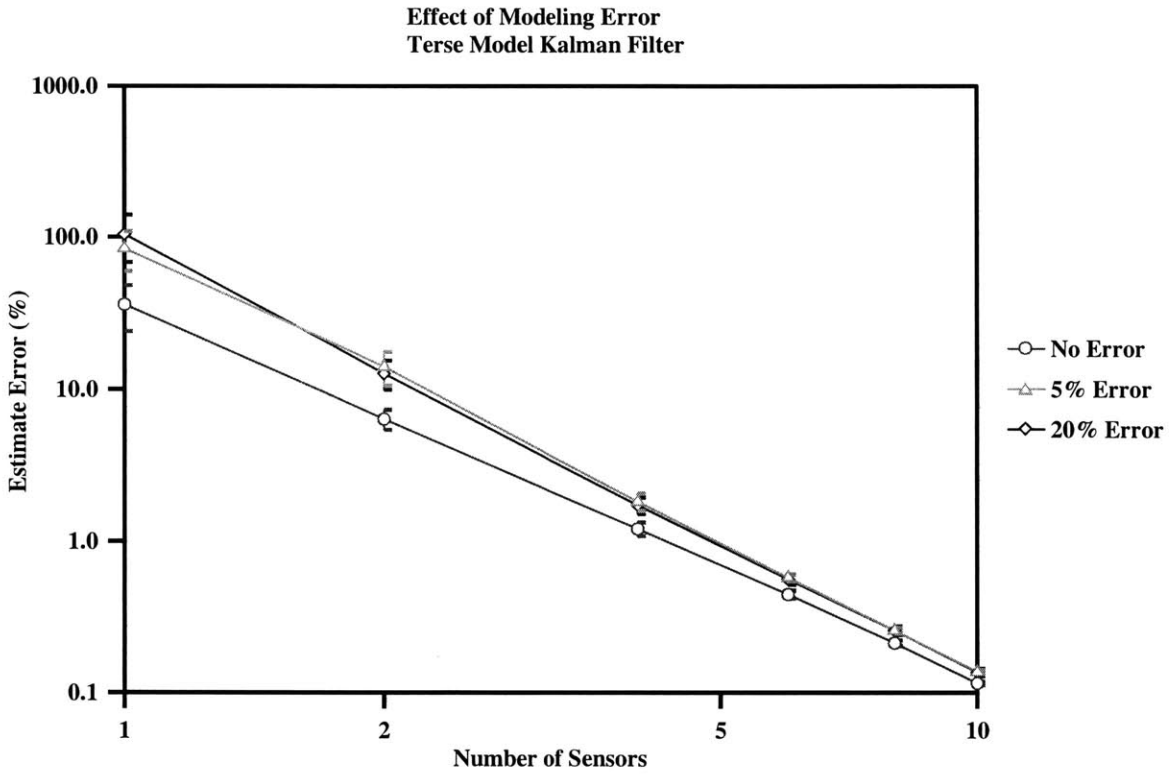


Figure 3-15: Effect of Modeling Error on Kalman Filter Estimation Error

Model Error	Number of Sensors					
	1	2	4	6	8	10
0 %	35.97	6.36	1.20	0.44	0.213	0.115
5 %	84.03	14.05	1.82	0.57	0.257	0.137
20 %	104.17	12.65	1.72	0.56	0.257	0.136

Table 3.7: Terse Kalman Filter, Effects of Model Error

Model Error	Number of Sensors					
	1	2	4	6	8	10
0 %	9.12	2.01	0.47	0.19	0.094	0.059
5 %	12.66	5.03	0.85	0.27	0.123	0.063
20 %	11.86	4.94	0.81	0.26	0.121	0.060

Table 3.8: Full Kalman Filter, Effects of Model Error

Although the Kalman filter is initially greatly effected by the change in the model, the results for large numbers of sensors are fairly similar. This is especially true after the introduction of sensor noise which raises the noise floor. As seen in Figure 3-16, the results seem to run together as the number of sensors increases.

The results of having included additional modes alleviates the modeling errors, as shown in Figure 3-17. Unlike the other version of the Kalman filter, this one shows much less increase for the single sensor case. The reason for this is that the one sensor case utilizes five modes instead of the typical doubling that is used for the cases with higher number of sensors. This means that in addition to reducing the amount of error due to aliasing, inclusion of additional modes also helps to compensate for modeling error.

3.6 Method Comparison

In the absence of errors the Kalman filter that includes multiple modes is clearly the best method, followed by the Quasi-Static with filter, and the Terse Kalman filter, with the unfiltered Quasi-Static method a distant last. Figure 3-18, shows the plot

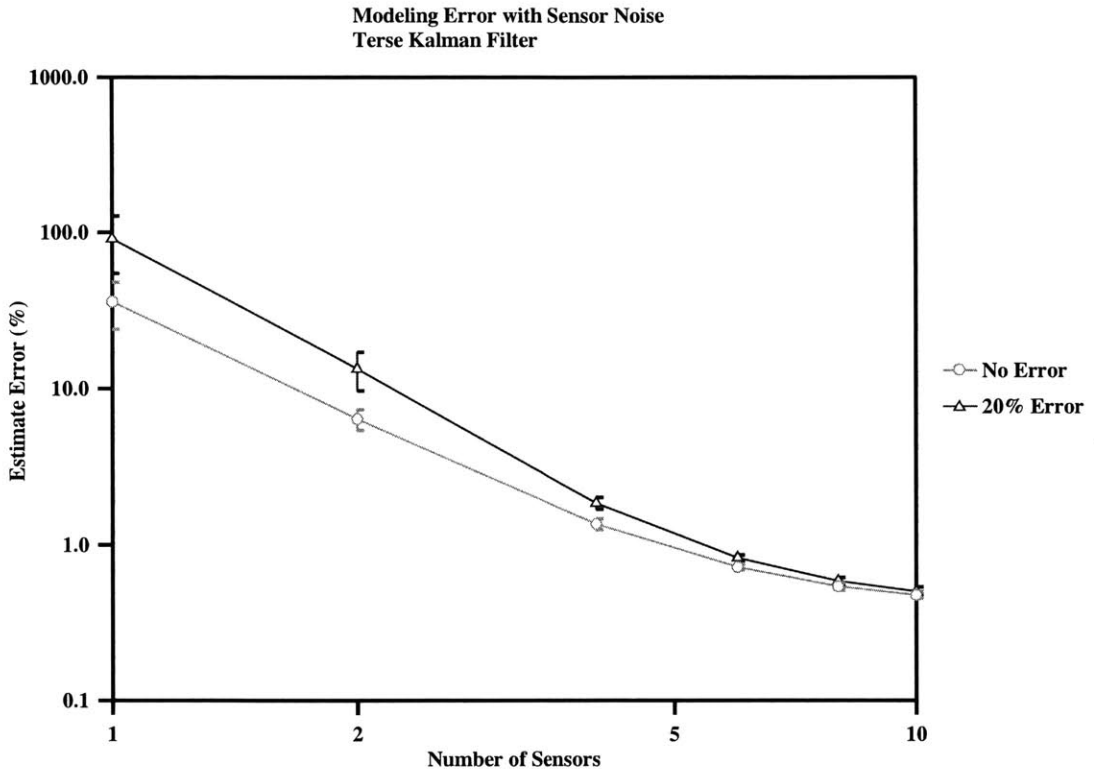


Figure 3-16: Effect of 20% Sensor Noise on Kalman Filter with Modeling Error

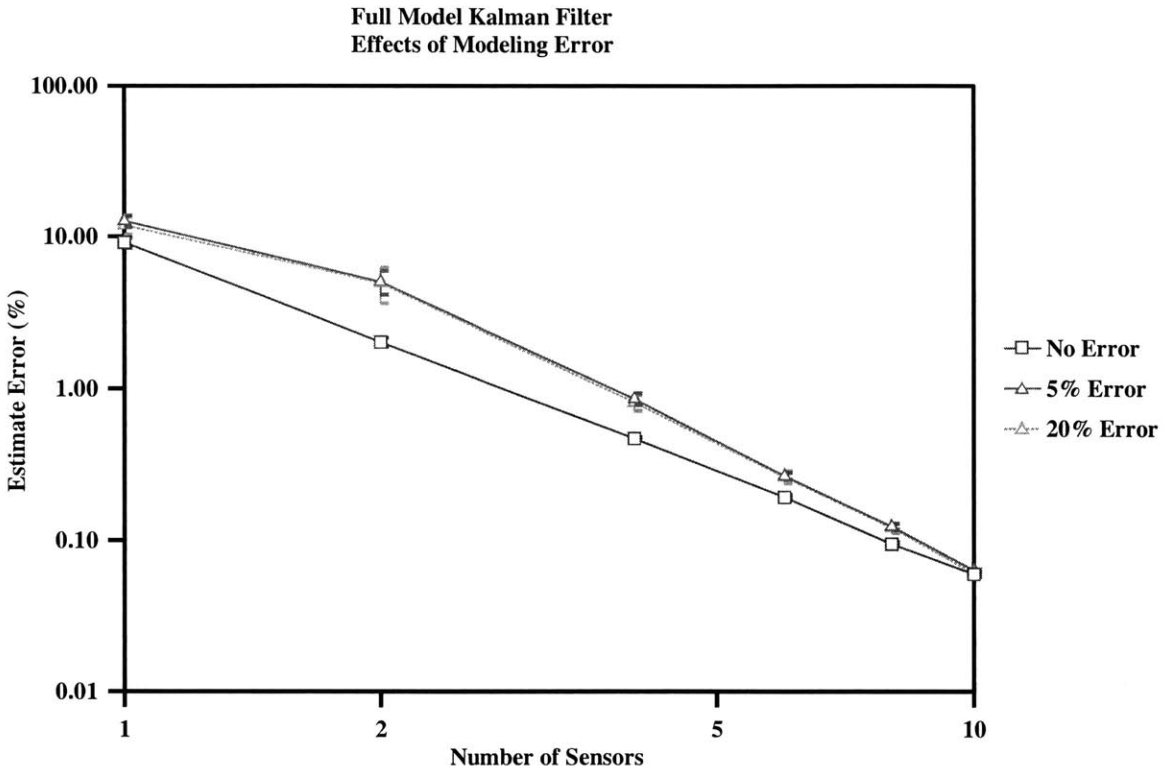


Figure 3-17: Effect of Modeling Error on Full Kalman Filter Estimation Error

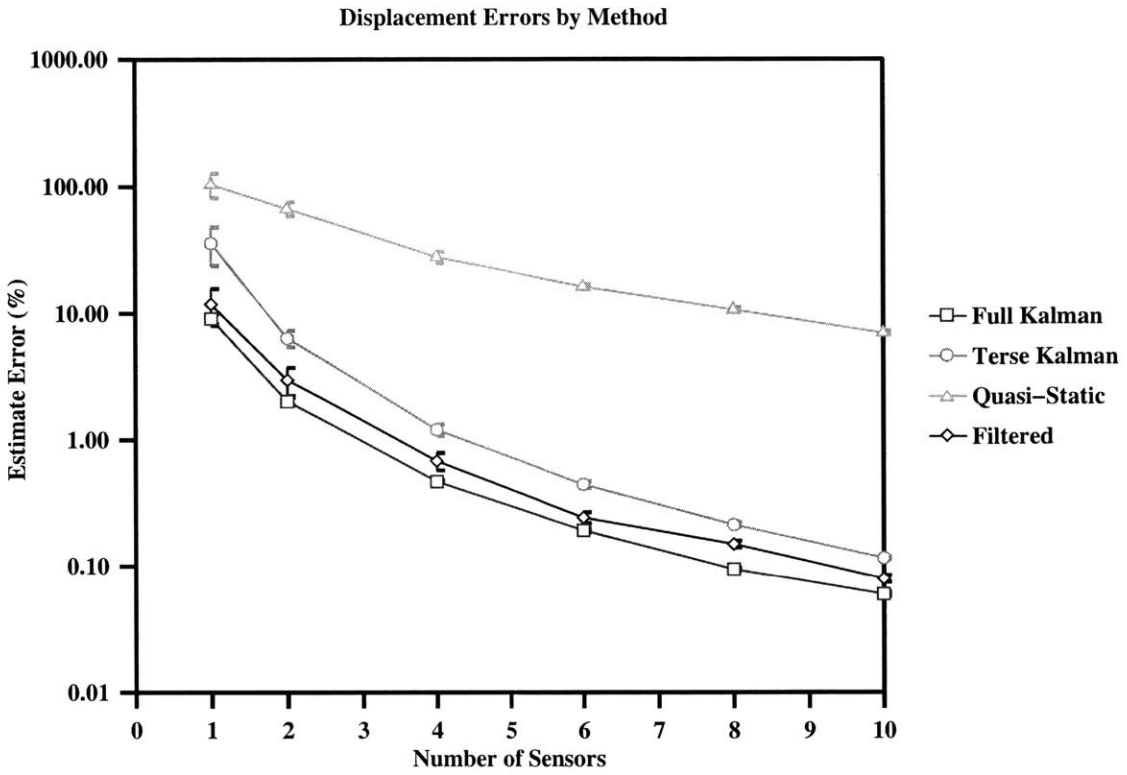


Figure 3-18: Comparison of Methods for Ideal System without Sensor Noise

of the errors for the system with no noise as a function of number of sensors. They are also in Table 3.6.

As the level of sensor noise increases, the Kalman filter remains the method of choice, regardless of the number of number of sensors. Figure 3-19, shows the error for each method as a function of sensor noise when there are 10 sensors.

The addition of modeling errors has the largest effect on the Kalman Filter, as the Quasi-Static methods, with and without filter, are largely unaffected, Figures 3-

Sensors	1	2	4	6	8	10
Quasi-Static	127.5	75.1	30.5	17.1	11.2	7.3
with Filtering	15.7	3.71	0.79	0.27	0.16	0.084
Terse Kalman Filter	48.0	7.32	1.33	0.47	0.22	0.12
Full Kalman Filter	9.8	2.15	0.48	0.20	0.097	0.062

Table 3.9: Estimation Error for Methods

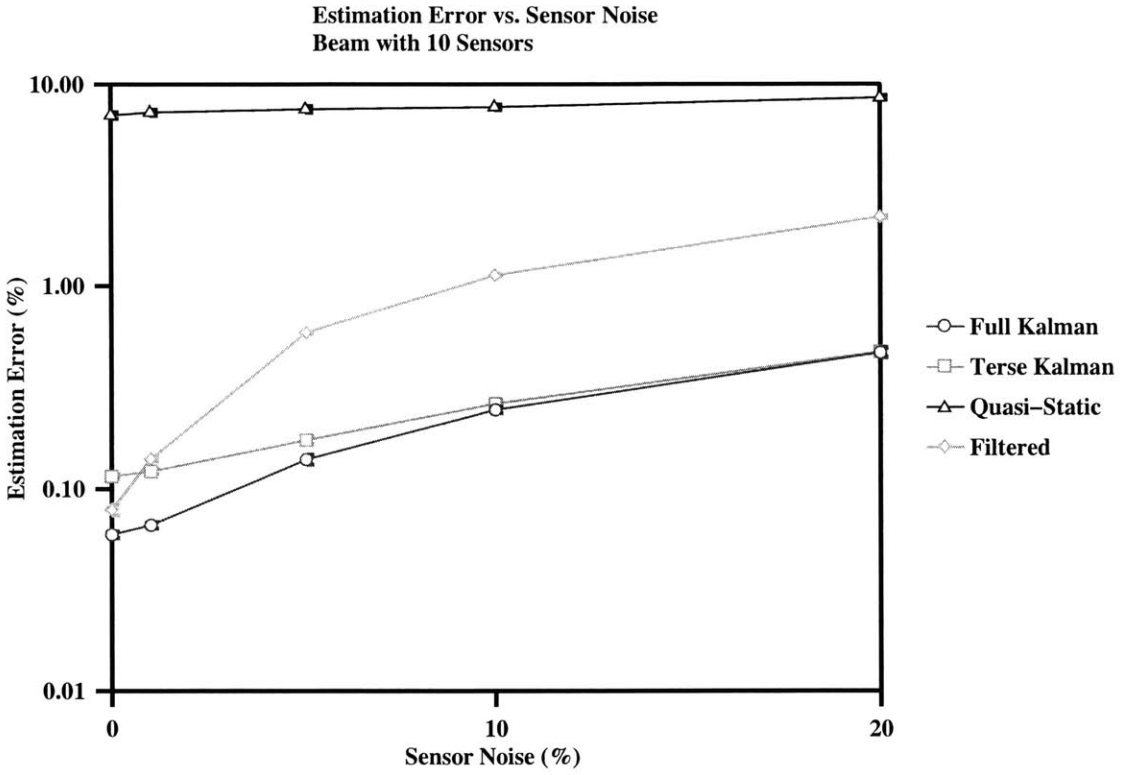


Figure 3-19: Model Behavior as a Function of Sensor Noise for Beam with 10 Sensors

7 and 3-10. This is especially true when the Kalman filter has few states. Small perturbations in the model cause large errors for the Kalman filter, fortunately larger levels of modeling error do not have progressively large effects. The Filtered Quasi-Static method however is less strongly influenced.

Additional results are available in Appendix A.

3.7 Summary

As expected the Kalman filter produces the lowest error in estimating the tip deflection. The Terse Kalman Filter does not perform nearly as well is the Full Kalman Filter, and in some cases not as well as the Filter Quasi-Static Method. All three methods perform better than the Quasi-Static Method.

Increasing the number of sensors clearly improves the effectiveness of all of the estimation methods. Having additional sensors allows a larger portion of the shape to be estimated, truncating less of the model. Additionally the Quasi-Static method benefits because the estimate can include a higher proportion of the strain energy of the structure, leaving less to be aliased.

Noisy sensors primarily affect the estimate for the higher modes of the structure. These modes have lower energy which is reflected by their lower strain readings. The lower readings are more easily lost in the noise floor. For the Filtered Quasi-Static this actually produces an increase in error when there are a large number of sensors. For the Kalman Filter the sensor noise results in the Terse and Full Kalman Models becoming equivalent.

Modeling error causes very little change in the performance of the Quasi-Static method, with and without filter. It does affect the Kalman Filter considerably however, especially when there are few sensors. Even with the additional errors causes by incorrect modeling the Kalman Filter produces the best results.

Chapter 4

Experiment

4.1 Experimental Setup

To obtain verification of the dynamic methods, a cantilevered beam was built and instrumented. The beam used was a steel ruler 18 inches in total length. The root-most three inches of the beam were fixed in a bench top clamp to provide the necessary boundary conditions. Measurements of the ruler show that it has a thickness of 40 mils, and a width of approximately 1 1/8 inches. The use of a ruler helped insure accurate placement of the sensors, and alignment of the clamp.

The sensors used were standard foil type strain gages, from Measurement Group, Inc., with gage lengths of 1/4 inch. The strain gages were centered at 3/16 inch, 3 inches, 6 inches, 9 inches, and 12 inches. The root strain gage was not able to be centered at the root. Attempting to do so would place the gage in the clamp which would result in erroneous readings, if the sensor worked at all. The position was chosen to be as close to the root as possible with the gage contacting the clamp. The positioning of the strain gages can be seen in Figure 4-1.

To sense the tip deflection of the beam, a Keyence laser displacement sensor was used. The visible spot of the laser was aligned to be 1/8 of an inch from the tip. This ensured that the laser remain on the beam at all times, and is close enough to the tip so as to accurately reflect the tip deflection.

The forcing for the beam came through the use of piezo-ceramic, PZT-5H. The

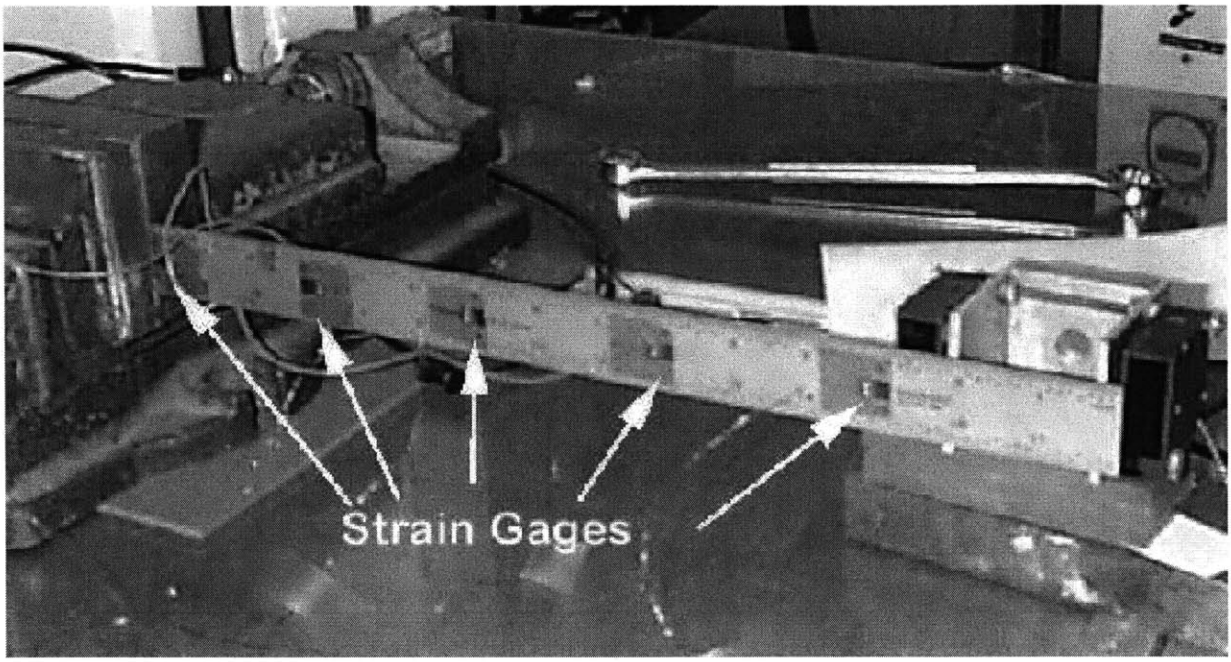


Figure 4-1: Photograph of the Experimental Setup Showing the Strain Gage Locations

piezo was sized to exactly fit the width of the beam, and to be 2.5 inches in length, with a thickness of 5 mils. The ceramic was placed 1/4 inch from the root of the beam. A piezo-ceramic was used for the excitation since it can provide high frequency forcing.

The piezo was modeled as moment couples located at the ends of the wafer. The positioning of the piezo and the way it transfers force as moment couples created more complicated mode shapes that the tip loading used in the simulation. Most notably the piezo was almost unable to force certain mode shapes, most noticeably, the fourth, when the wavelength of the structure is nearly equal to the wavelength of the piezo.

A picture of the experimental setup showing the piezo and laser vibrometer is shown in Figure 4-2. A measured drawing which better shows the locations of the sensors is shown in the diagram in Figure 4-3. The piezoelectric is attached on the opposite side of the beam from the strain gages.

The data collection and transfer function calculations were done using two Siglab signal analyzers, Model 20-42 made by DSP Technology Inc. A total of eight streams of data were collected. There are the five strain gage signals, the laser displacement

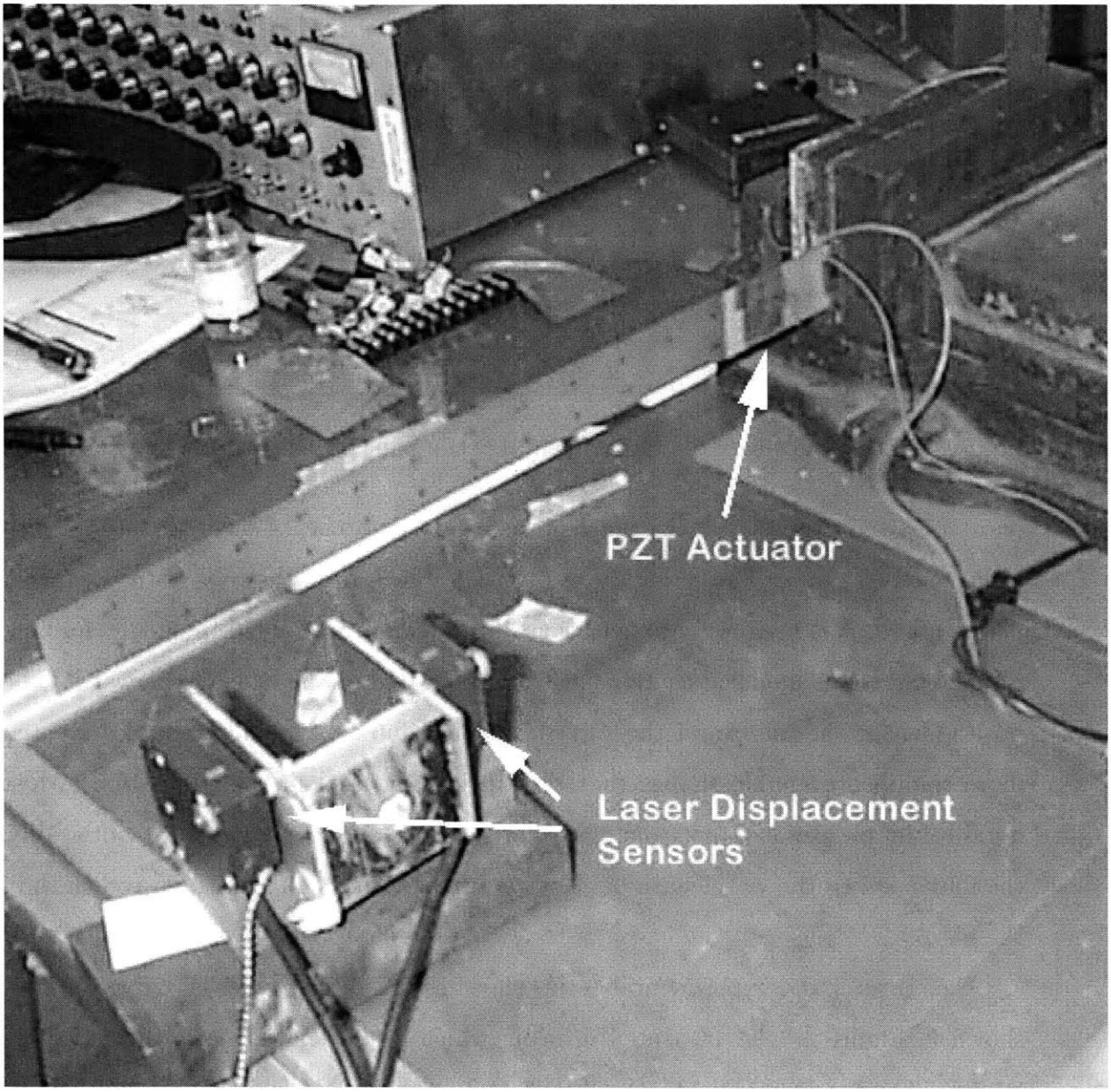


Figure 4-2: Photograph of the Experimental Setup Showing the Laser Vibrometer and PZT Actuator

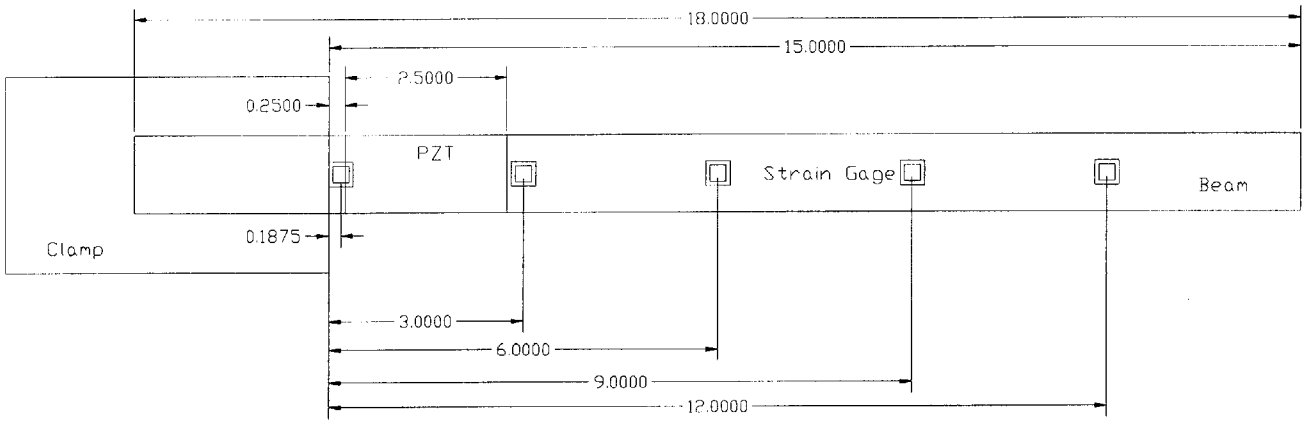


Figure 4-3: Measured Drawing of Experimental Setup

sensor signal, the output of the amplifier and the excitation signal generated by Siglab. The output from Siglab and the amplifier were compared to determine the amplifier dynamics.

The voltage source used to run the experiments was a Trek amplifier Model 622, with a power supply Trek Model 663A, which was used to amplify the white noise signal produced by SigLab. The amplifier has a double pole at 214 Hz. The transfer function of the amplifier is shown in Figure 4-4. Since the amplifier has dynamics in the frequency range of testing, the output is not white noise.

4.2 Model Determination

Shape estimation methods require an estimate of the strain readings for each gage for each of the modes of the structure. For an experimental setup there are two ways to do this. The first method is to use an analytic tool, such as a Finite Element Method or Rayleigh-Ritz analysis, which can give good approximations. The accuracy of these methods is dependent on the complexity of the structure, and the accuracy of the model.

The second method, and the one used in this experiment, was to determine the transfer functions for each of the sensors, including the laser which measures deflection [19, 12, 10]. For the Quasi-Static method the transfer function is not strictly

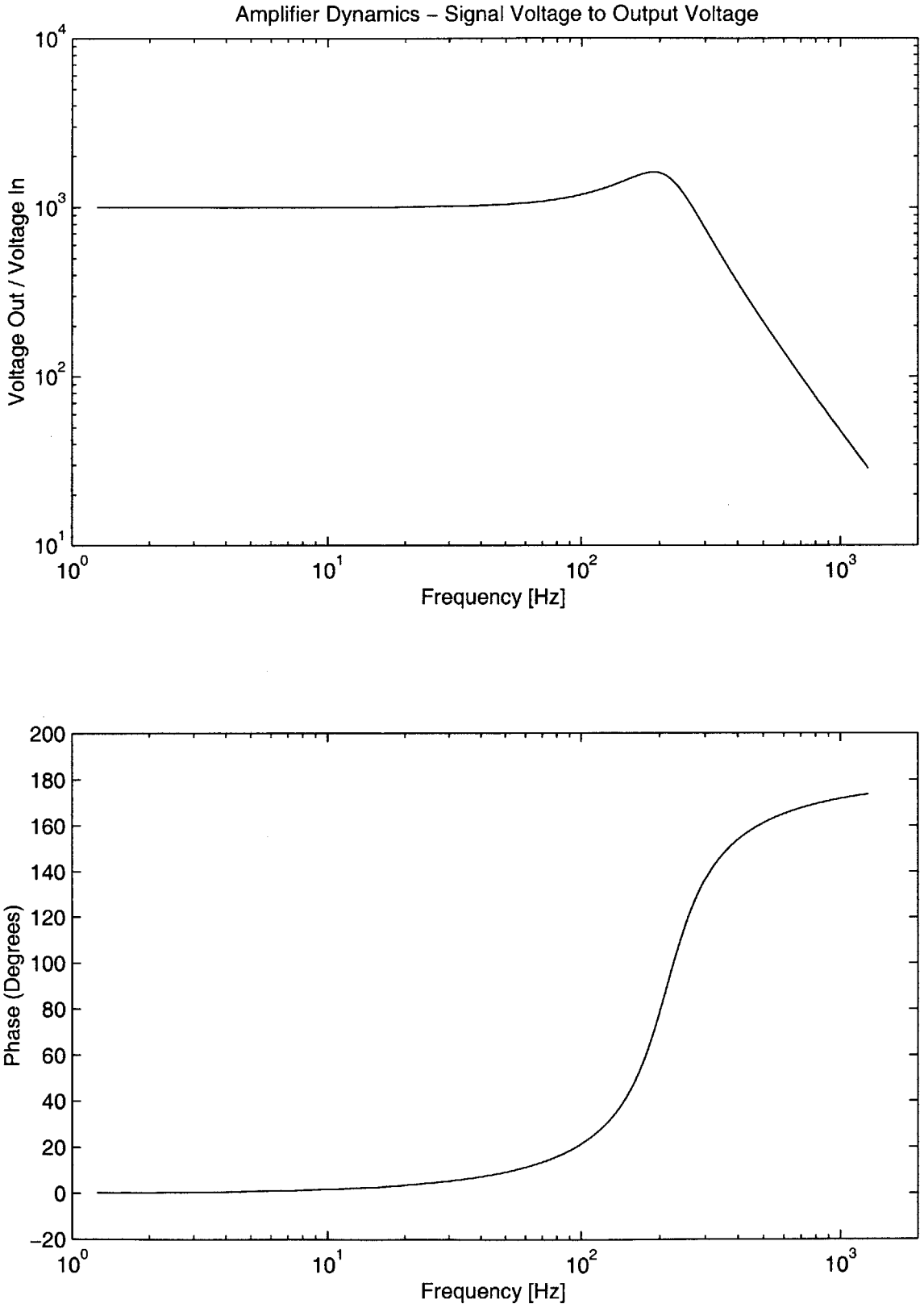


Figure 4-4: Transfer Function showing Amplifier Dynamics

necessary, but the modal residues provide the best way to find the modal weights for each strain gage.

To fit the state space model the first step was identifying the modal frequencies. This can be done from any of the transfer functions. The half-power bandwidth is used to determine the damping ratio for each mode. Once the damping ratios and modal frequencies are determined the A matrix can be fixed.

The next step is to examine the transfer function from applied voltage to the tip deflection. Since it is simplest to normalize each of the modes by the tip displacement, the C matrix for the displacement is set to unity. The coefficients of the B matrix, or forcing matrix, can then be fit to the tip deflection transfer function by matching the peak amplitudes.

Once the A and B matrix are determined it is a simple matter of finding the remaining coefficients of the C matrix, for each strain gage. Later this C matrix will directly correspond to the H matrix used in the Kalman filter, as seen in Equation 2.9.

The overall shape of the structure was not experimentally determined for this test, since the experiment was aimed at estimating the tip deflection only. For general deflection knowledge a number of approaches are available. One method is to use a finite element, or similar technique, to obtain estimates of the modes shapes and frequencies. The shapes can then be scaled by referencing to individual points such as the tip of the beam, since the displacement shapes do not vary greatly. Another technique is to find the displacement transfer functions at multiple points and do a spline interpolation for the overall shape.

4.3 Results

4.3.1 Model

The transfer functions for the strain gages and the laser displacement sensor were found using Siglab. To increase the resolution, the bandwidth was divided into sections and concatenated. The bandwidths were 0-10 HZ, 10-50 Hz, 50-150 Hz, 150-350

Hz, 350-550 Hz, 550-850 Hz and 850-950 Hz. The bandwidths were chosen to provide ample points at each mode to fit the empirical model.

Figure 4-5 shows the transfer function from the output voltage of the amplifier to the laser displacement sensor. At around 140 Hz the laser shows an unexpected mode, and more unexpected modes at higher frequencies. These modes are the torsion modes of the beam, which are greatly amplified by the type of laser sensor used. Since the displacement reading is sensitive to the slope of the surface that is read.

The strain gage transfer functions are much cleaner than that of the laser displacement sensor. For the most part, they do not show any indication of the torsional modes of the structure, which is because the strain gages are insensitive to shear. At higher frequencies the zeros of the structures are harder to pick out, particularly for the tip-most strain gage. The transfer functions for the strain gages can be found in Figures 4-6– 4-10.

At high frequencies the output from the amplifier is so attenuated that the zeros get damped out. The result is that the zeros do not show up clearly on the amplitude plot and the phase is much slower to respond. At the breaks between frequency sweeps there are slight discontinuities in the phase and in some cases amplitude due to the anti-aliasing filter used by Siglab.

The first eight modes were clear enough to be modeled effectively. Higher modes than that were too close to the noise floor to allow a good fit, due in large part to the amplifier's roll-off. Table 4.3.1 shows the resulting modal frequencies and damping ratios.

After determining the frequencies and damping ratios the gain matrices can be determined. The gains for the forcing matrix, B matrix, are determined by the amplitude of the tip deflection at each frequency. The sensitivity matrix, C matrix, is determined by examining the ratio of the displacement transfer function to the strain gage transfer function for each gage at each frequency. Table 4.3.1 shows the resulting gains. The displacement is the ratio of the laser signal voltage to actuation voltage. The strain gage gains are the ratio of the strain gage output voltage to the laser output voltage. These are the actual gains used in the shape estimation.

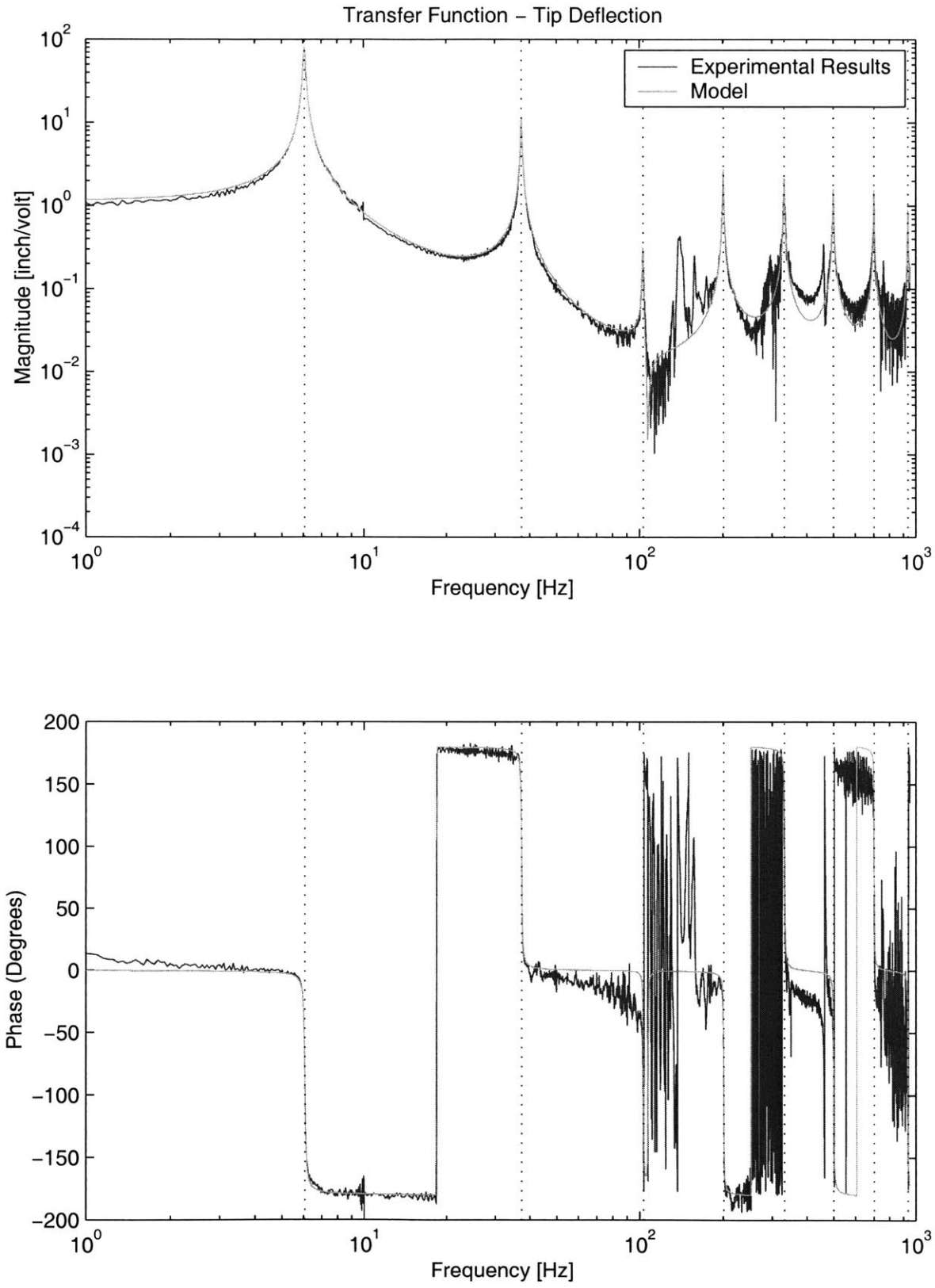


Figure 4-5: Experimental vs. Modeled Transfer Functions for Tip Deflection

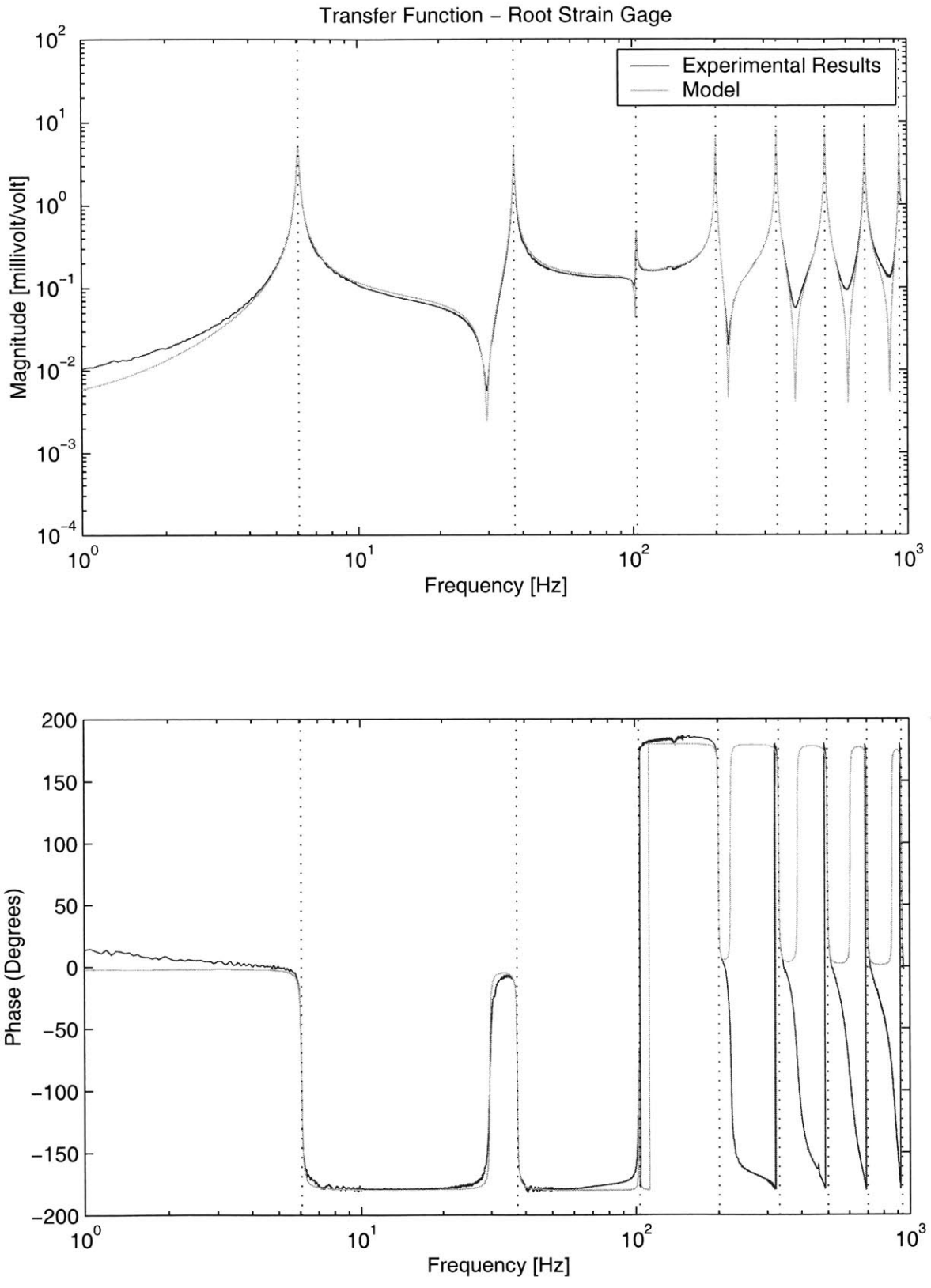


Figure 4-6: Experimental vs. Modeled Transfer Functions for Root Strain Gage

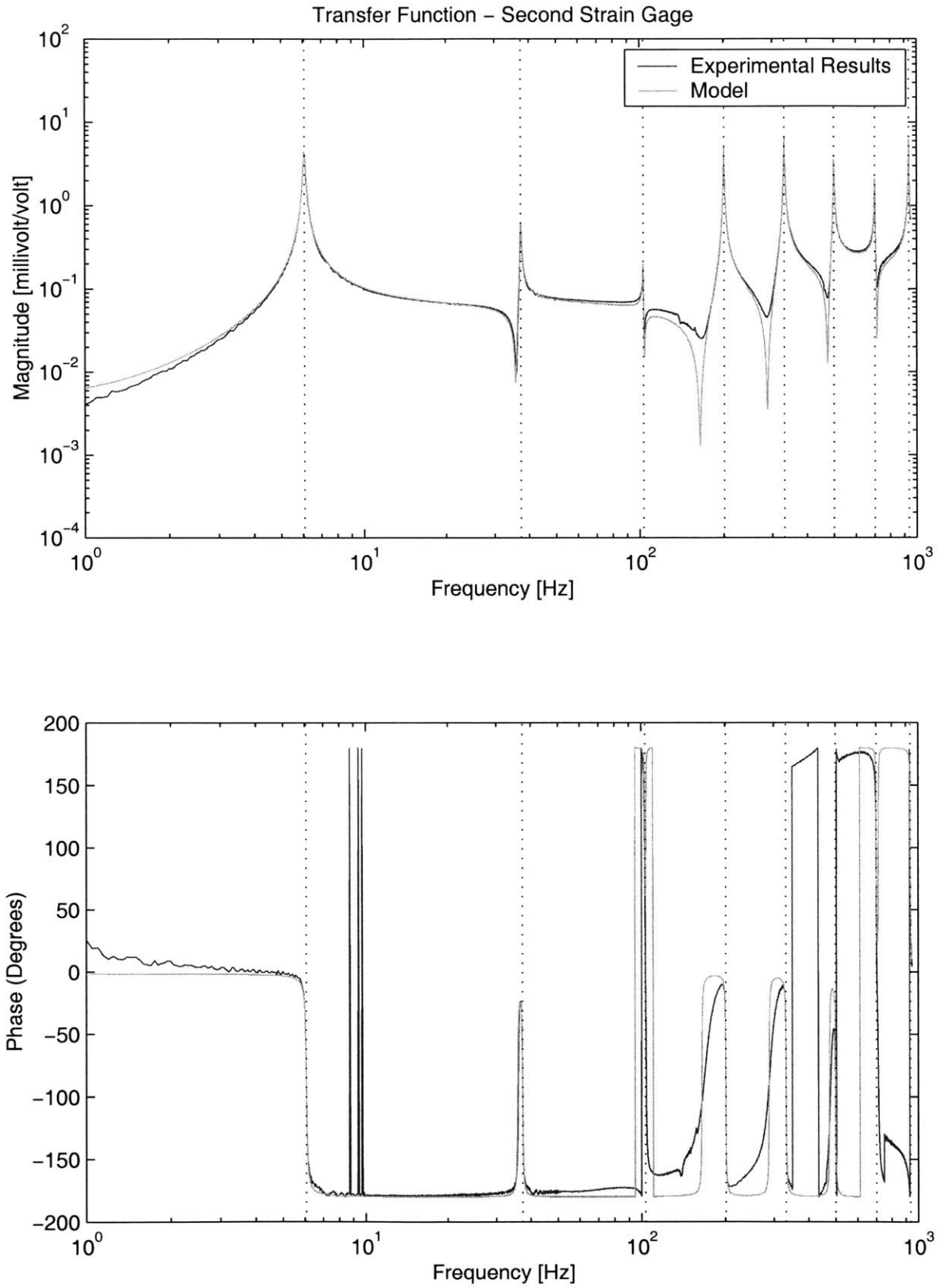


Figure 4-7: Experimental vs. Modeled Transfer Functions for Second Strain Gage

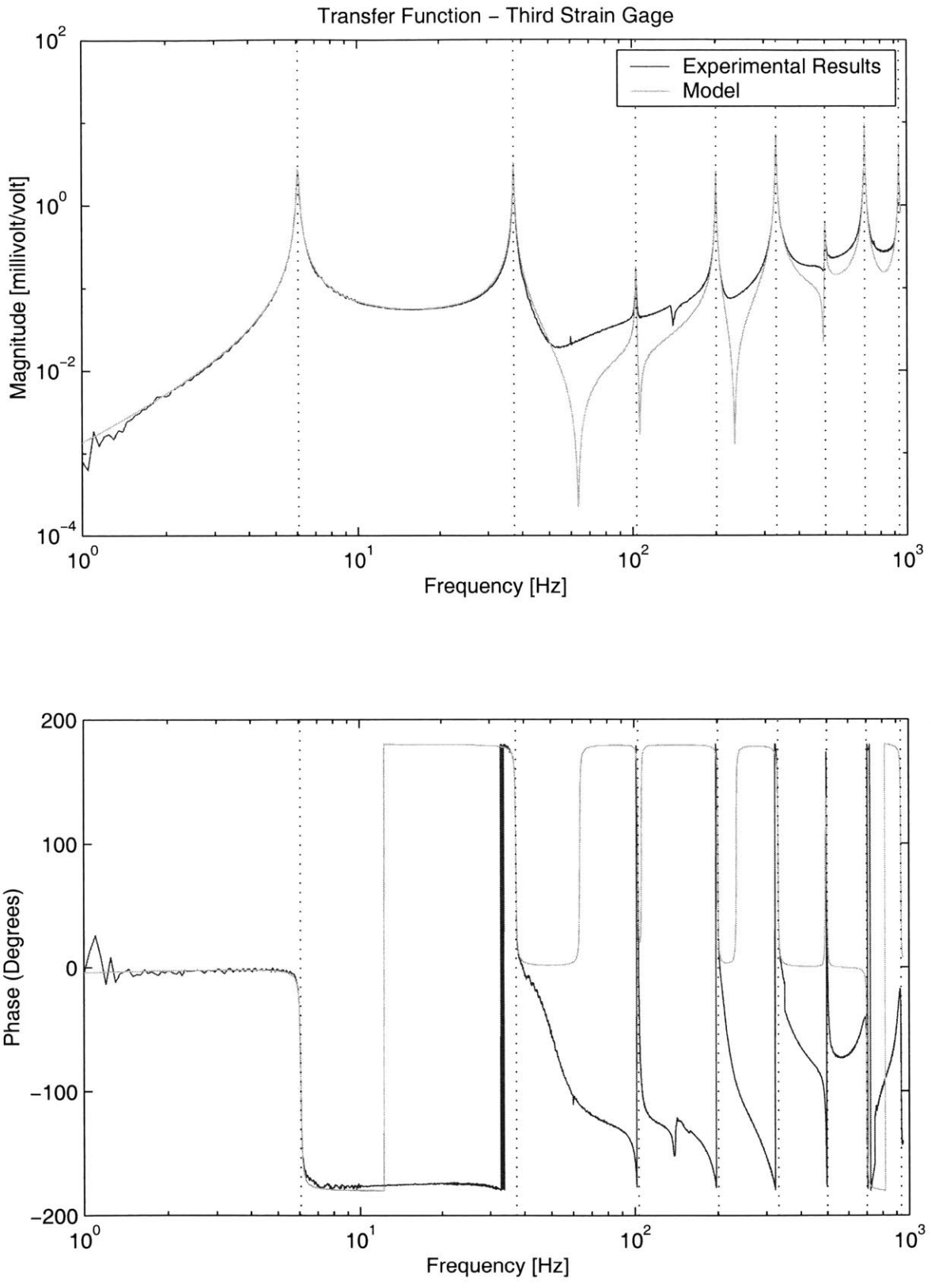


Figure 4-8: Experimental vs. Modeled Transfer Functions for Third Strain Gage

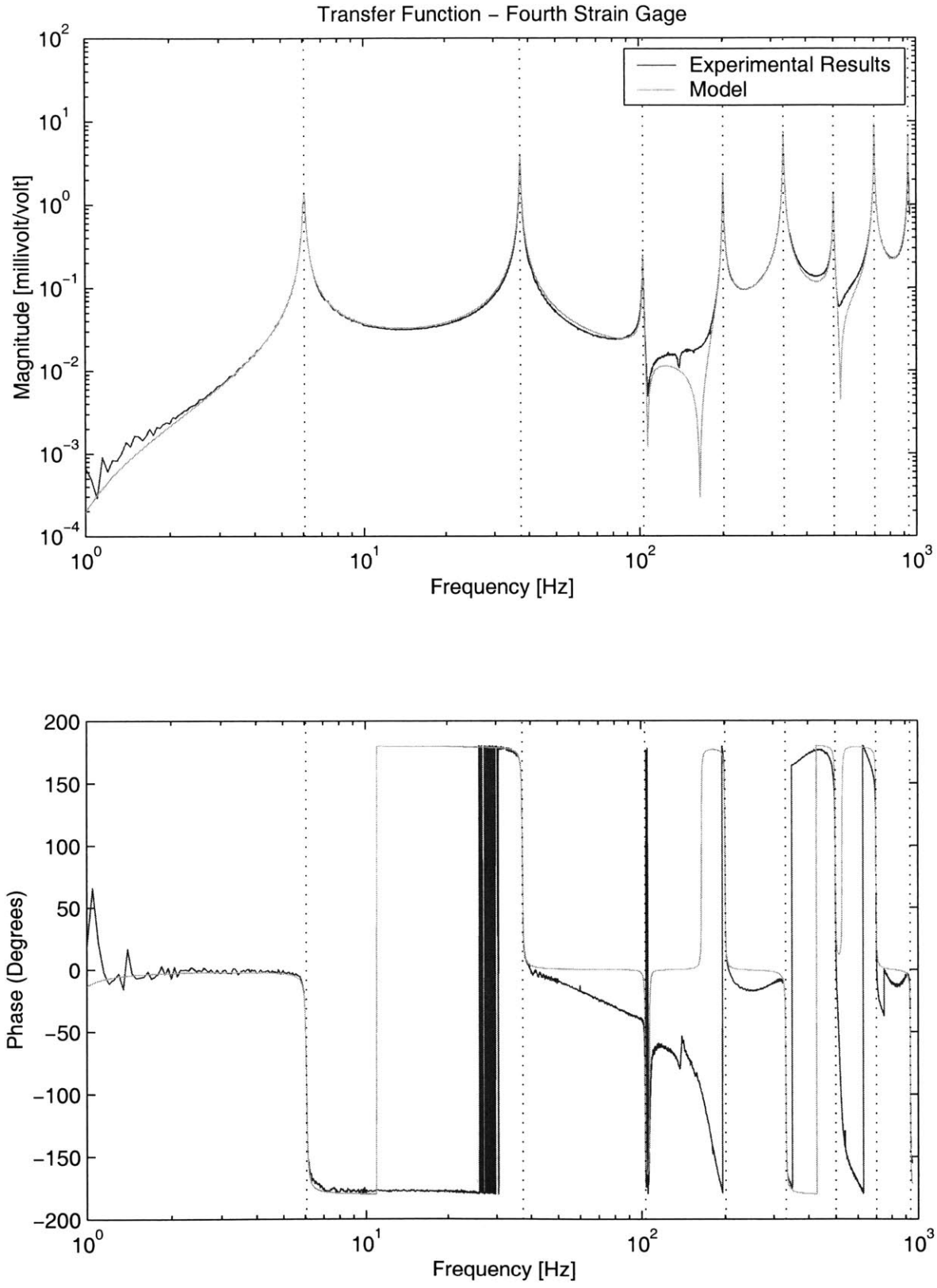


Figure 4-9: Experimental vs. Modeled Transfer Functions for Fourth Strain Gage

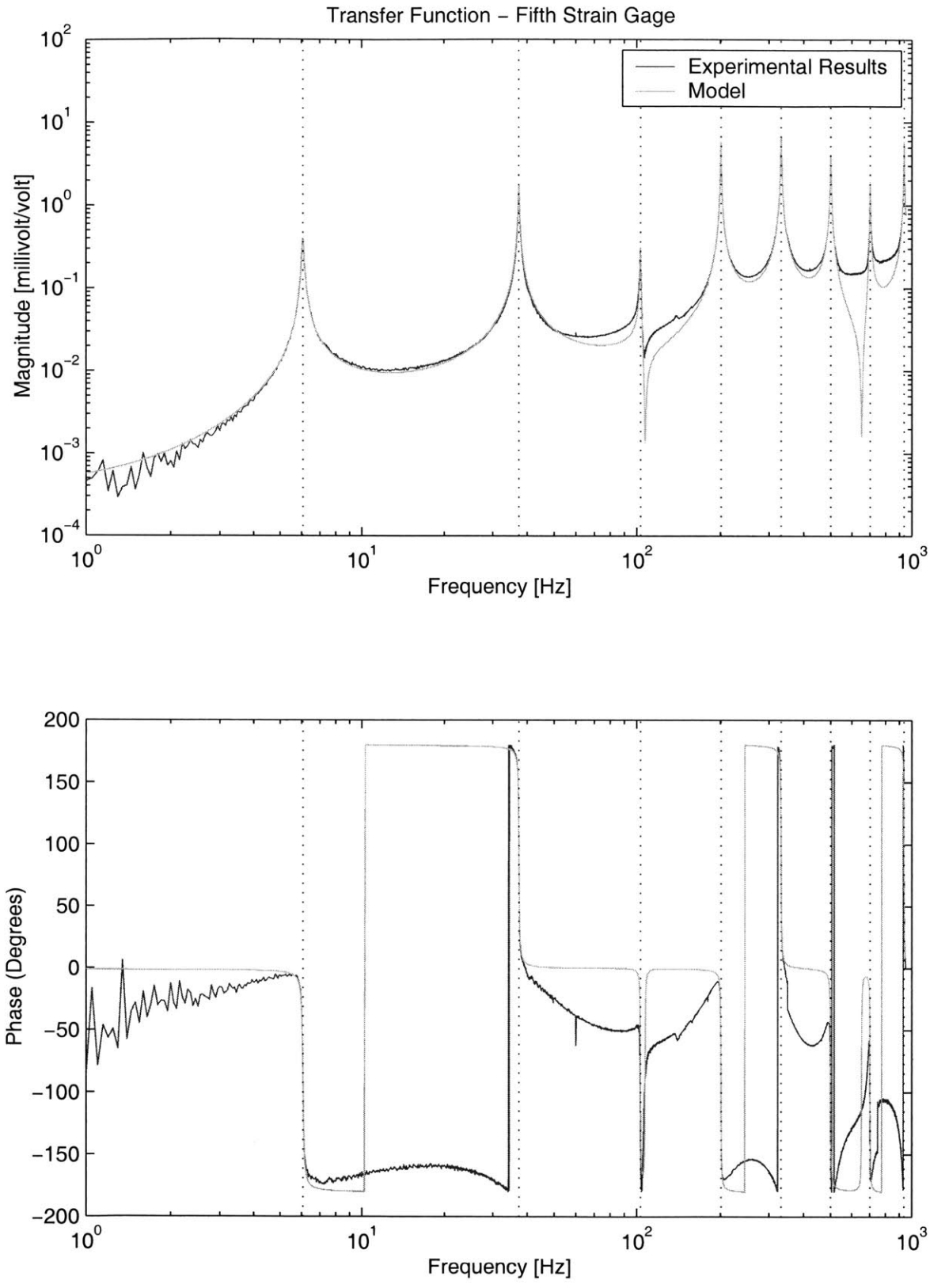


Figure 4-10: Experimental vs. Modeled Transfer Functions for Fifth Strain Gage

Mode Number	Frequency (Hz)	Zeta
1	6.1	0.0070
2	37.4	0.0045
3	103.1	0.0026
4	200.8	0.0025
5	332.0	0.0030
6	500.5	0.0030
7	702.0	0.0020
8	936.8	0.0020

Table 4.1: Modal Frequencies and Damping Ratios

Mode Number	1	2	3	4	5	6	7	8
Displacement	35.00	-4.50	0.13	1.15	-0.85	0.63	-0.6	0.35
Strain Gage 1	0.16	-1.15	3.60	-5.60	9.20	-12.89	16.00	-21.00
Strain Gage 2	0.14	-0.14	-1.44	4.70	-7.50	5.70	3.50	-18.00
Strain Gage 3	0.09	0.75	-1.45	-2.30	8.50	-0.95	-16.20	-16.30
Strain Gage 4	0.04	0.88	1.90	-2.15	-8.60	-2.25	16.00	20.00
Strain Gage 5	0.01	0.40	2.40	5.20	8.00	6.40	-3.15	-16.4

Table 4.2: Experimental Gains

It is important to note that the third mode has a very low displacement gain. Physically this is an artifact of the position and size of the piezo-ceramic, causing it to have low actuation authority for this mode. It should also be noted that the strain gage gains increase dramatically with the mode number since for a higher mode to obtain equivalent levels of deflections the local strains must be very high. But although the gains are higher, the actual magnitude of the strain remains roughly constant with frequency.

Figure 4-11 shows the locations of the strain gages on the predicted strain modes. The strain modes were calculated using the same Finite Element routine used in the simulations. Although the magnitudes of the strain gage readings tend to vary from the predictions, the general values remain close. The discontinuities in the strain shape are from the piezo-ceramic which adds considerable local stiffening.

4.3.2 Experimental Results

Once the model was fully determined the experiment could begin. The experiment consisted of running the system with white noise for 64 seconds and collecting data at the rate of 2560 Hz. The data collected included the five strain gages, the laser displacement sensor and the output voltage from the amplifier.

The data was imported into Matlab to be run through the shape estimation code, Appendix D. The code performs the estimation using the standard Quasi-Static method, two forms of filtering the Quasi-Static method and the Kalman Filter based method with 5 modes and 8 modes.

The first filter used on the Quasi-Static estimate was a 25 coefficient FIR digital filter with a cutoff frequency of 407 Hz. The frequency was chosen as the geometric mean of the fifth and sixth modes, which are the last sensed mode and the first unsensed mode respectively. The filter was applied to estimated deflection.

The second Quasi-Static filtering method uses separate filters for each mode, with a cutoff frequency dependant on the natural frequencies of the system, and then sums the result. The cutoff for the first mode is between the first and second mode, the cutoff for the second mode is between the second and third mode, etc. This

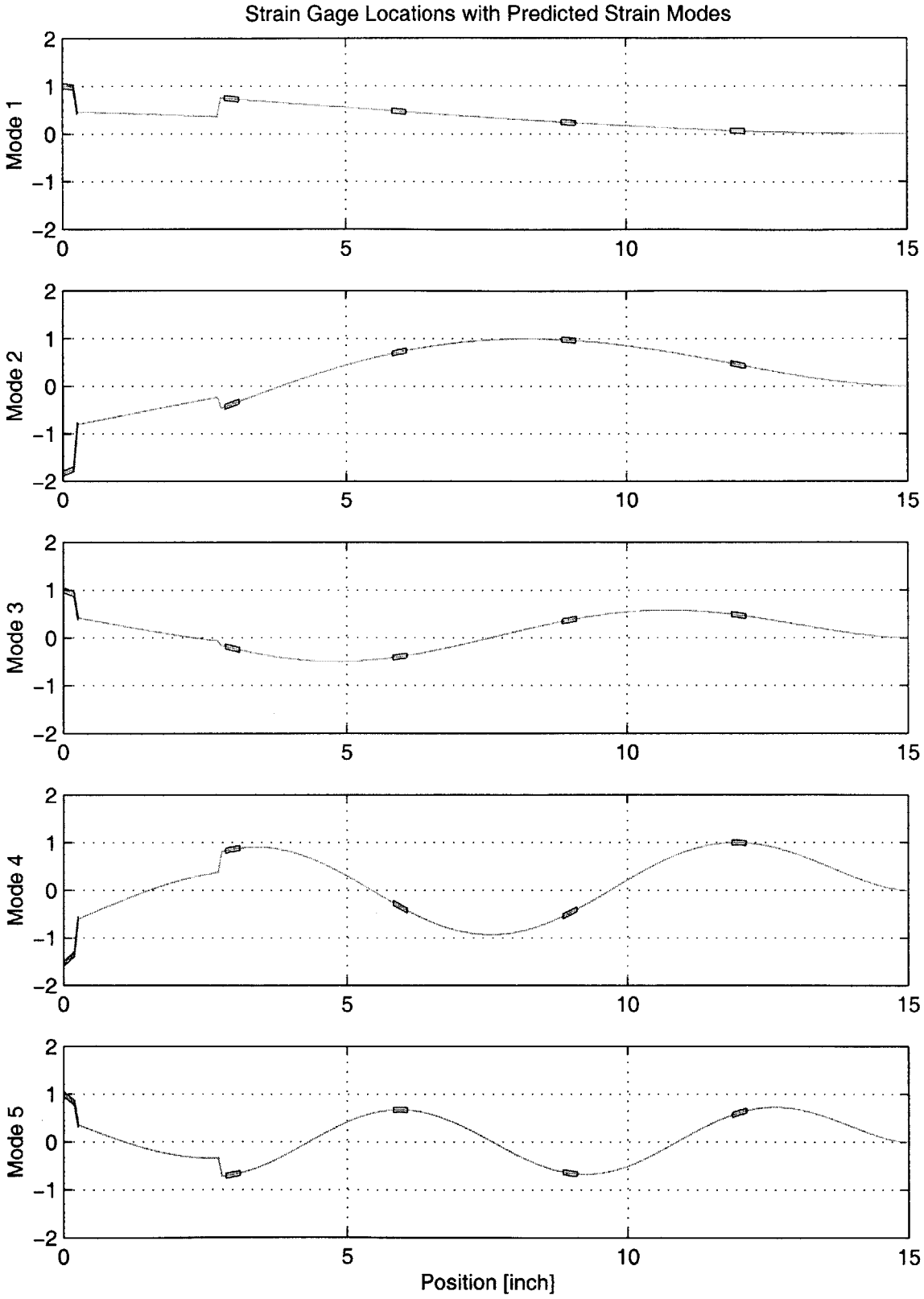


Figure 4-11: Sensor Locations on Predicted Strain Modes

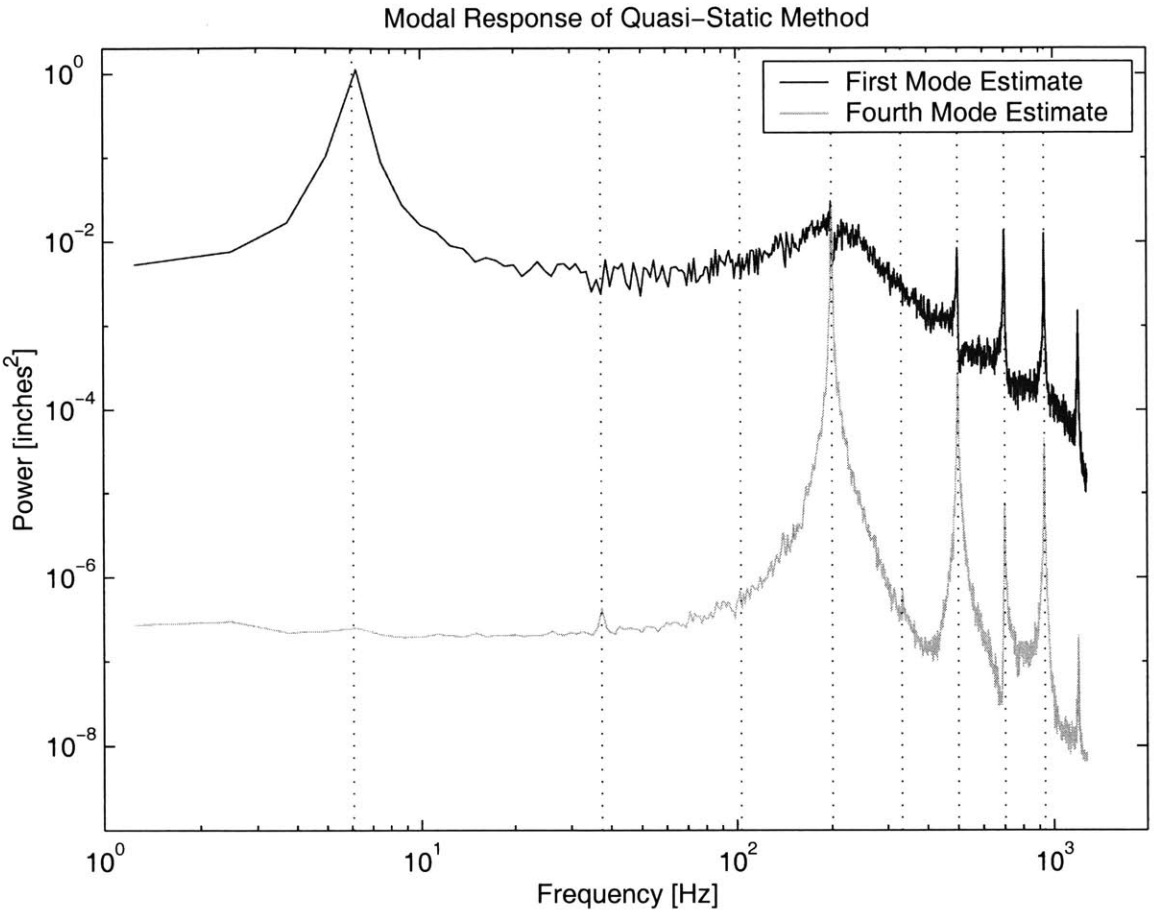


Figure 4-12: Estimated Modal Amplitudes using Quasi-Static Method

method will be referred to as the Modally Filtered Quasi-Static (MFQS) method. This method was used because it was found that each mode has its own noise floor that is proportional to the gain of the mode. The estimated response of the first mode dominates total response as shown in Figure 4-12. The fourth mode's peak amplitude is barely greater than the noise produced by the first mode.

Two different models were used to run the Kalman Filter. The first model contains only information on the first five modes of the structure. This is the same information that the Quasi-Static methods have available. The second model includes a total of eight modes, since only the first eight modes of the structure are clearly discernable.

Method	Error (%)
Quasi-Static	81.3
Filtered QS	74.0
Modally Filtered QS	26.5
5 Mode Kalman Filter	16.3
8 Mode Kalman Filter	16.7

Table 4.3: Experimental Results

After finding the estimated deflections using each of the techniques the error was calculated. The final error for each method is the root mean square estimation error normalized by the root mean square measured deflection. The errors are shown in Table 4.3.2. The first few seconds were not included in the error calculations to allow the Kalman filter to recover from not having an initial estimate of the modal amplitudes.

The time histories of the estimated displacements show more clearly what levels of errors each method produces. Figure 4-13 shows the time response of the Quasi-Static method on the top plot, and the error on the bottom plot. Figure 4-14 shows the Filtered Quasi-Static estimate of the response and the error.

A large improvement is shown in Figure 4-15 which shows the Modally Filtered Quasi-Static method. Further improvement is shown using the 8 mode Kalman Filter method as shown in Figure 4-16.

4.4 Analysis

It is clear that the Kalman Filter does the best job of estimating the tip deflection of the beam, followed by the Modally Filtered Quasi-Static method. It is also clear that using the Quasi-Static method will produce greater errors in the estimate of position that assuming no deflection.

The reason for this becomes more clear when examining the power spectral densities of the resulting tip deflection estimates. Figure 4-17 shows the PSD for the Quasi-Static method on the top half and the resulting error on the bottom half. Af-

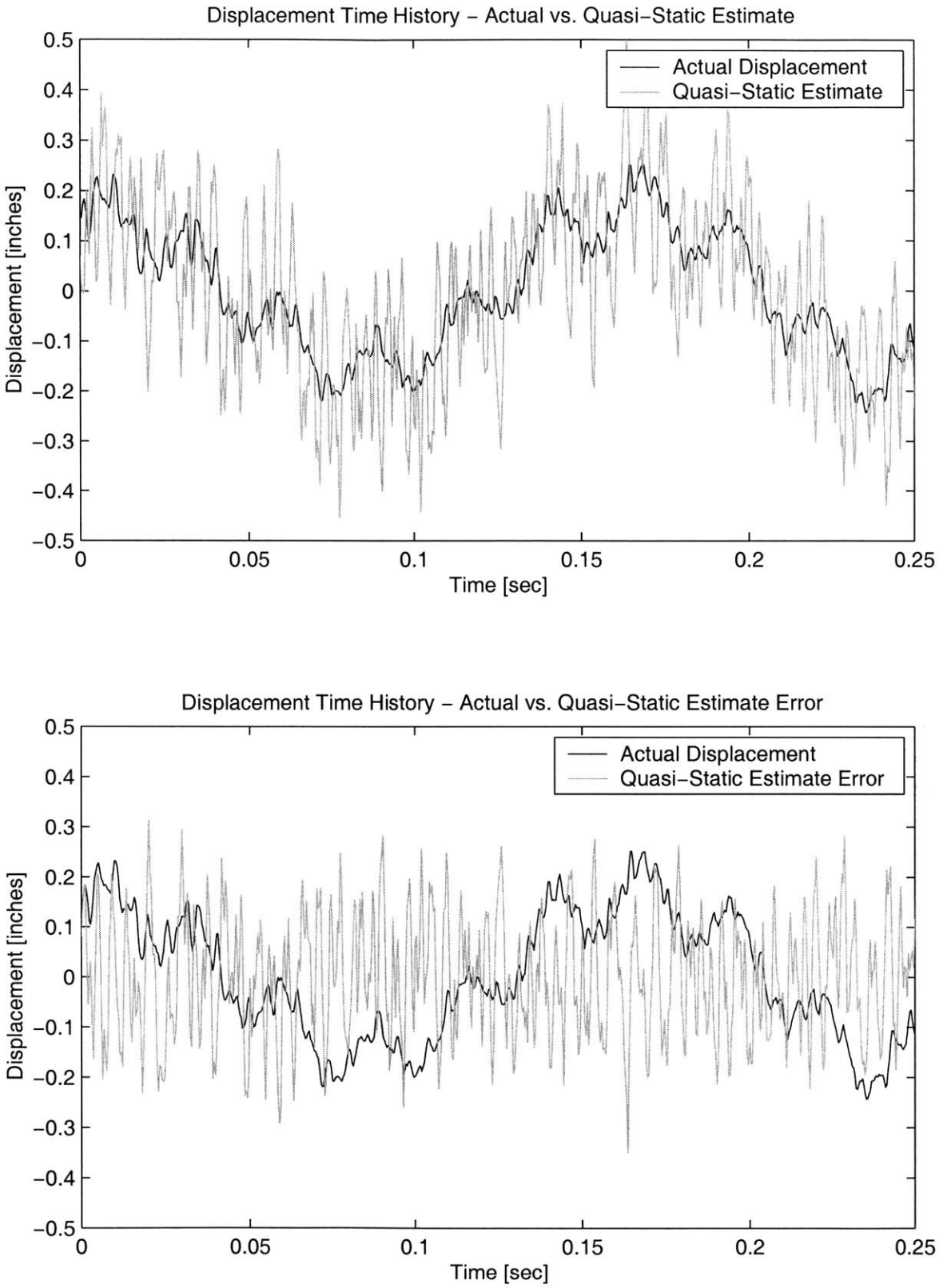


Figure 4-13: Time History of Quasi-Static Estimate vs. Actual Displacement

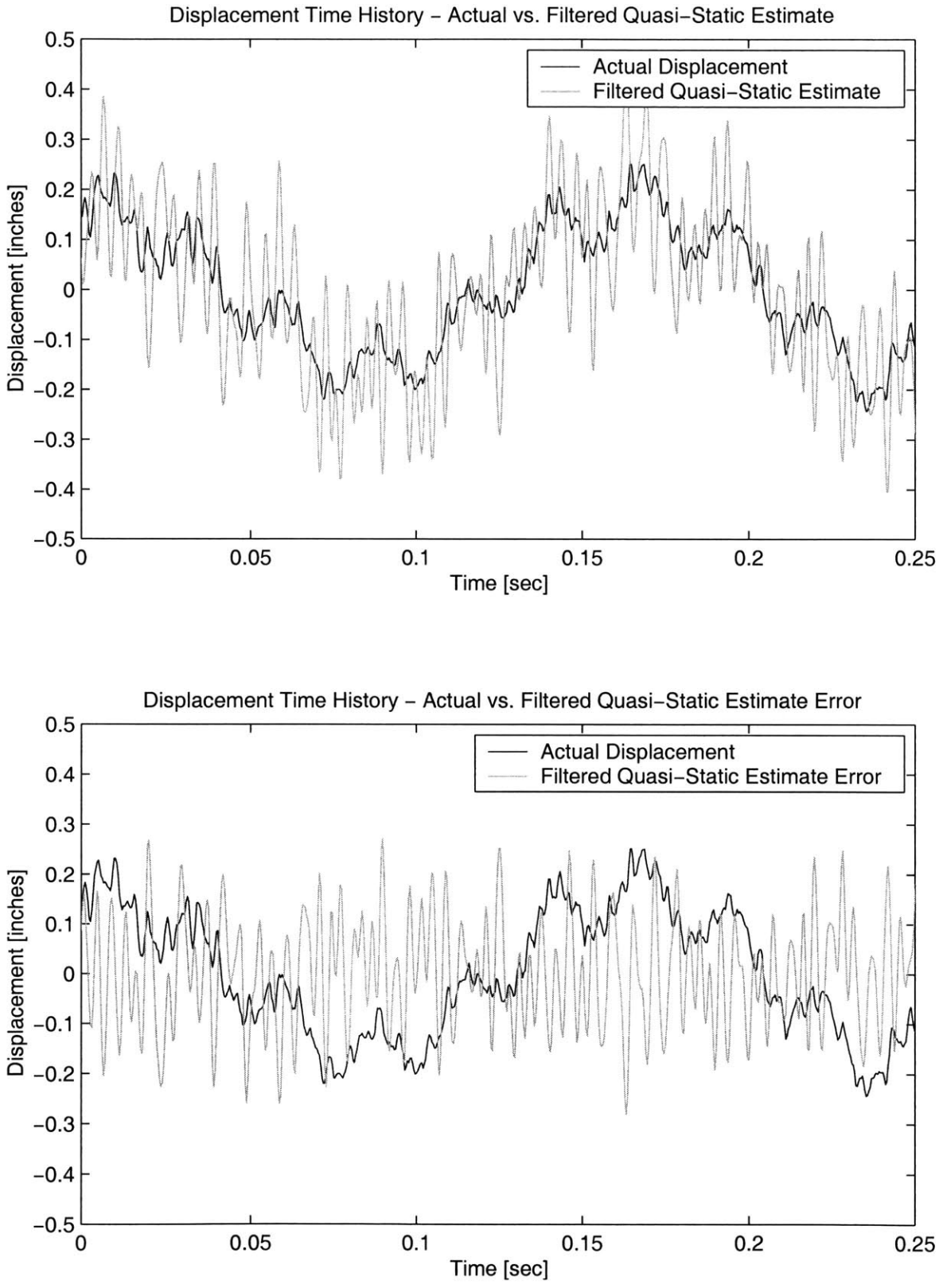
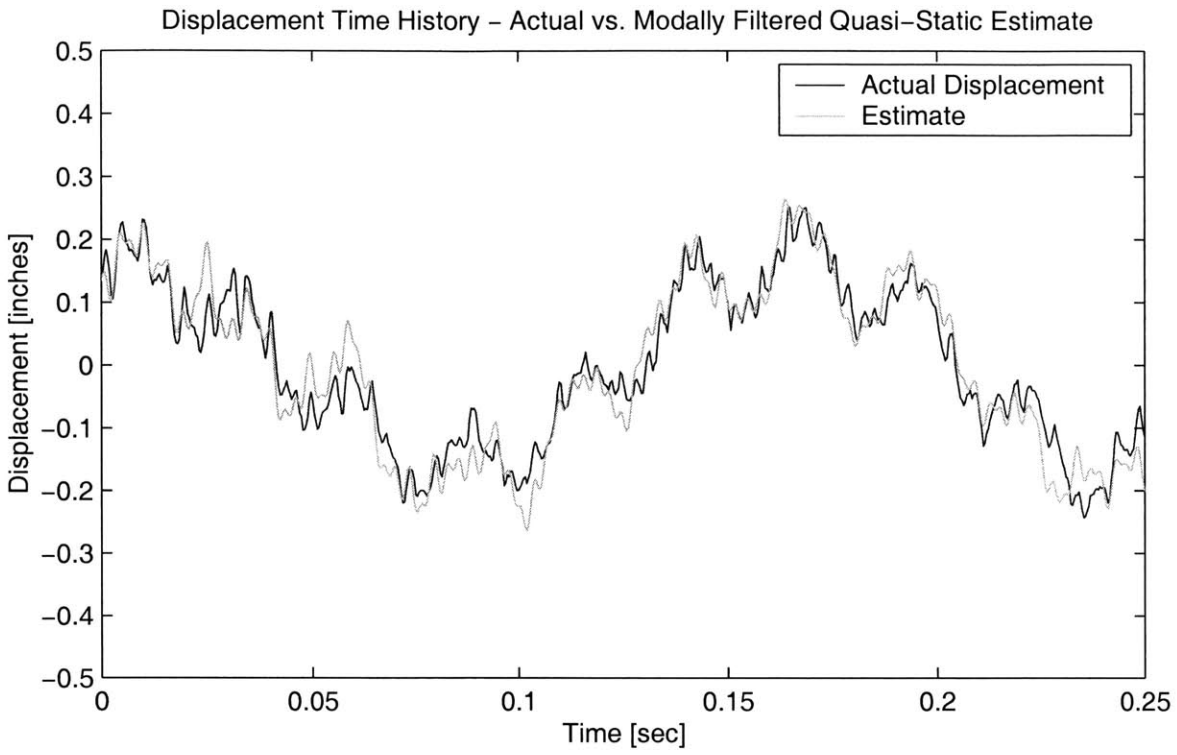


Figure 4-14: Time History of Filtered Quasi-Static Estimate vs. Actual Displacement



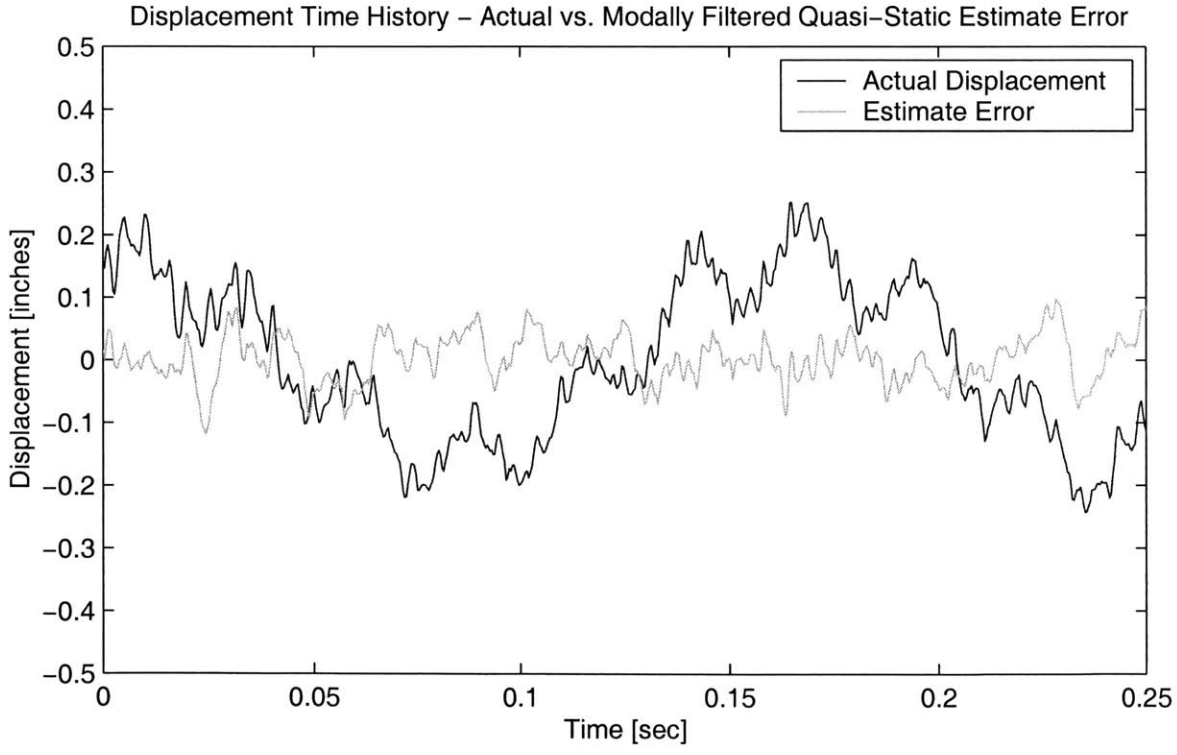
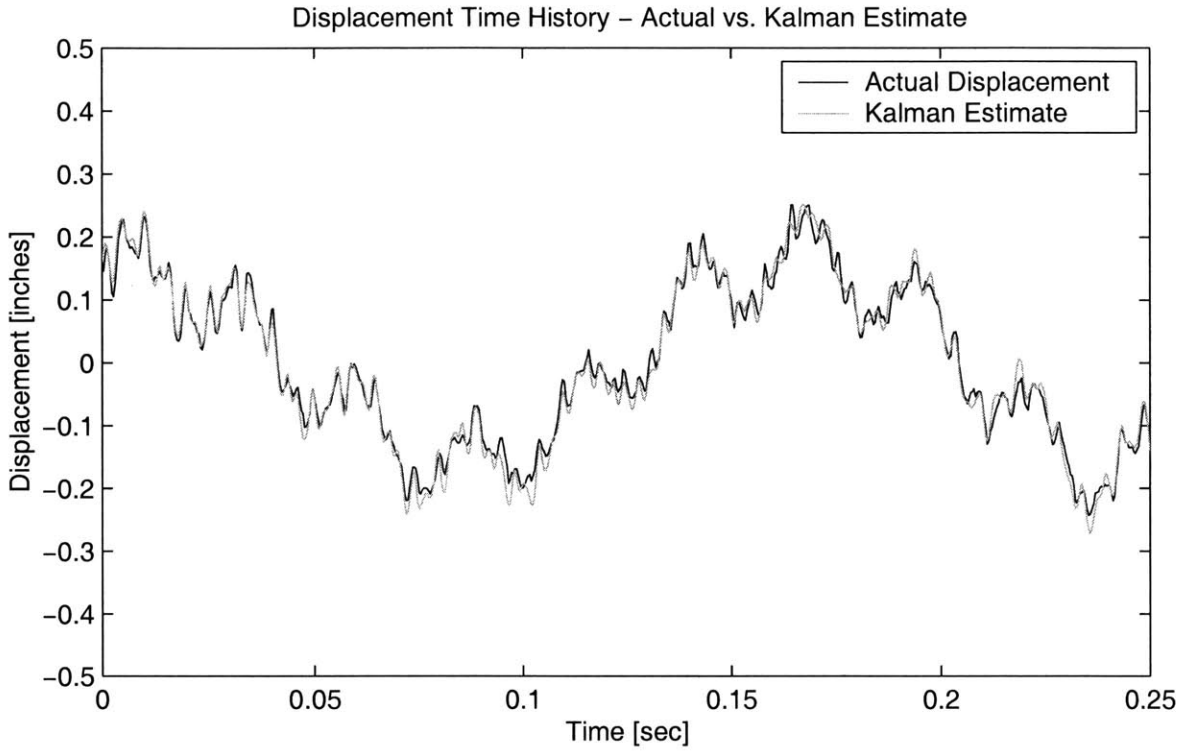


Figure 4-15: Time History of Modally Filtered Quasi-Static Estimate vs. Actual Displacement



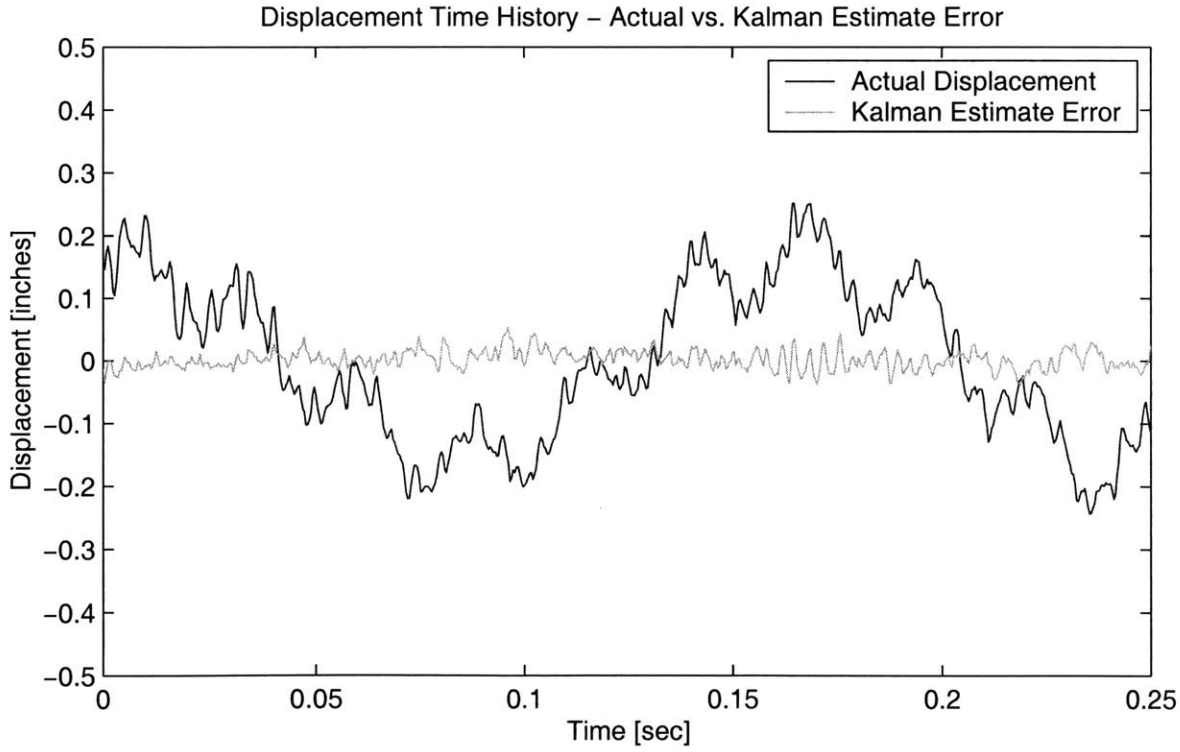


Figure 4-16: Time History of the 8 Mode Kalman Filter Estimate vs. Actual Displacement

ter the second modal frequency the estimated deflection is all error. The rounded peak around the fourth mode and subsequent roll-off in error are a result of the amplifier dynamics. In a system with true white noise the peaks of the sixth, seventh, etc. modes would actually be as high or higher than the peak for the first mode as was the case in the simulations.

Figure 4-18 shows the results of adding the FIR filter between the fifth and sixth modes. The low frequency results are almost identical to the unfiltered case which is to be expected. The filter does, however, successfully filter most of the high frequency noise. The high frequency noise being the peaks at the sixth, seventh, etc. modes caused by aliasing.

There is still the high noise floor caused by the first mode, which is allowed to persist until after the fifth mode. Figure 4-19 demonstrates the advantage of using a different temporal filter for each mode. The estimate at the fourth and fifth mode is no longer dominated by the estimated first mode response. This results in much less estimation error.

The Kalman filter does a much better job than even the MFQS method, as shown in Figure 4-20 and Figure 4-21. The difference between the 5 mode Filter and the 8 Mode filter are actually very slight. The higher modes are simply too close to the noise floor to be accurately sensed.

At very low frequencies the Kalman Filter performs slightly worse than the three other methods. This is largely because the Kalman filter is incapable of dealing with a DC offset in force as it is implemented. The 5 mode Kalman Filter actually performs better at low frequencies than the 8 mode Kalman Filter, while after the first mode the 8 mode Kalman Filter generally performs better.

The improved performance of the 5 mode Kalman Filter at low frequencies is most likely due to aliasing occurring in the 8 mode Kalman Filter. At low frequency the static response of each mode dictates the total response. The 8 mode Kalman Filter can no longer rely on frequency information to distinguish between modes and it becomes an indeterminate problem. The 5 mode Kalman filter is trying to fit only 5 modes to the 5 sensor readings which is a deterministic problem.

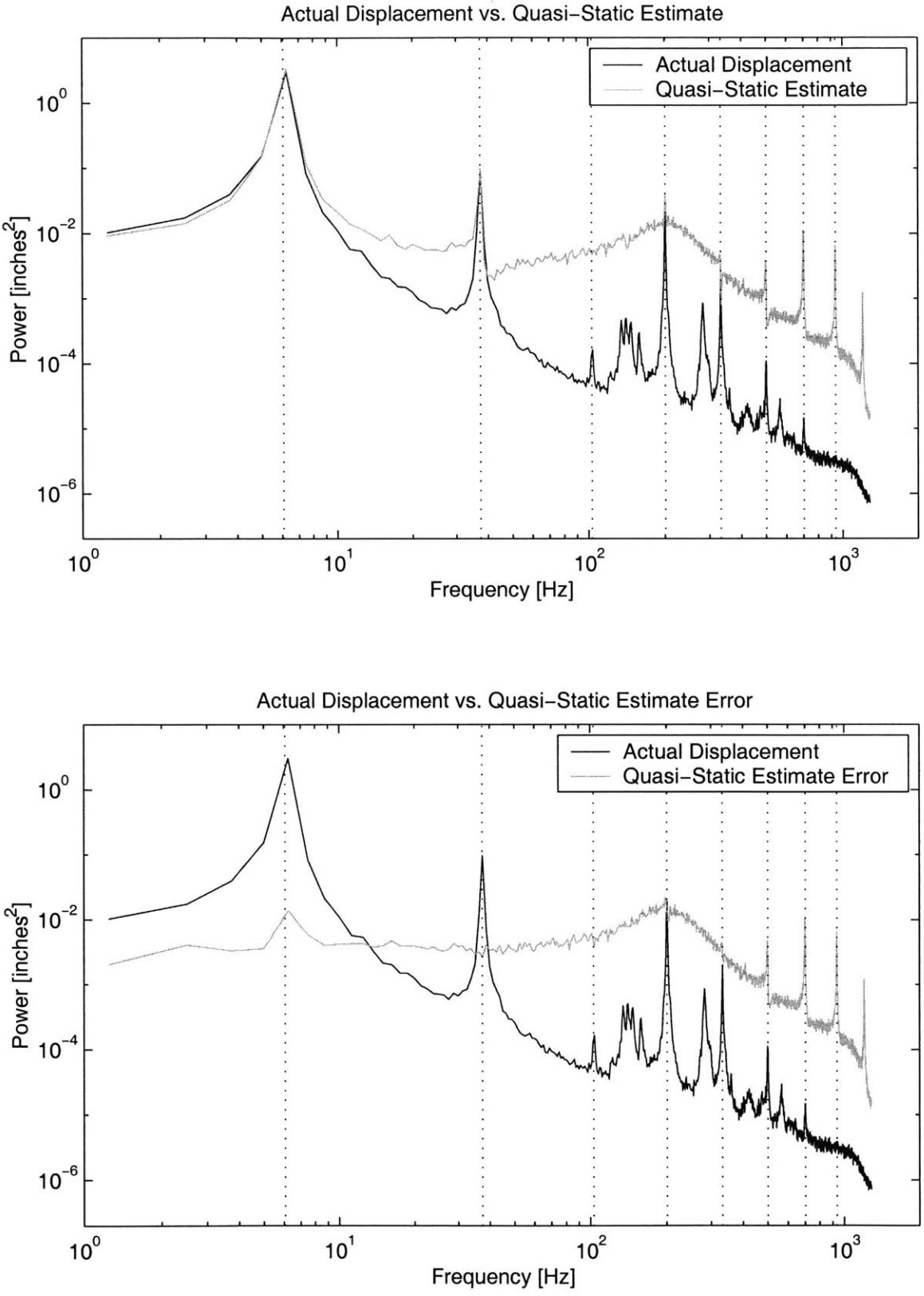


Figure 4-17: Power Spectral Density of the Quasi-Static Estimate vs. Actual Displacement

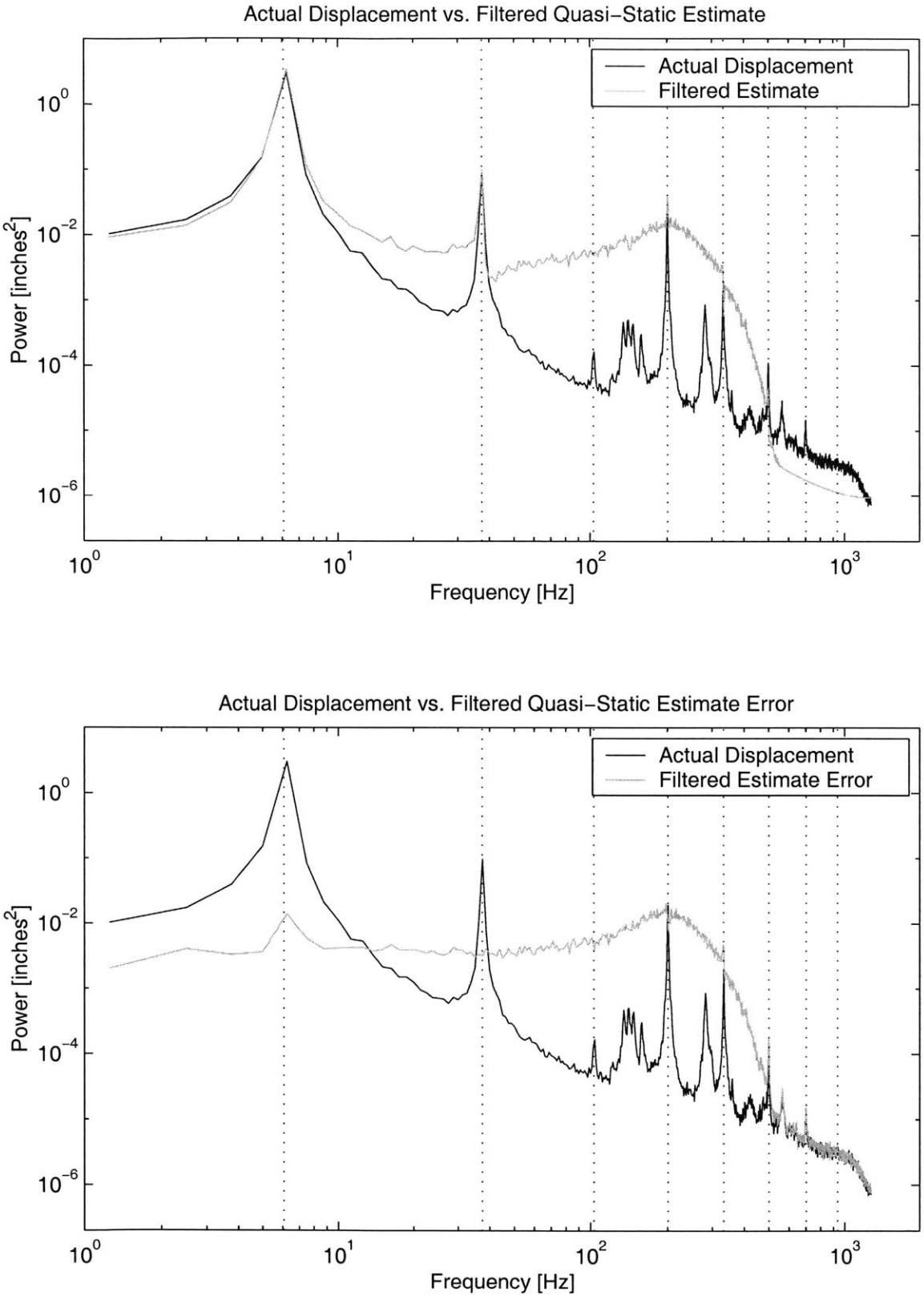


Figure 4-18: Power Spectral Density of the Filtered Quasi-Static Estimate vs. Actual Displacement

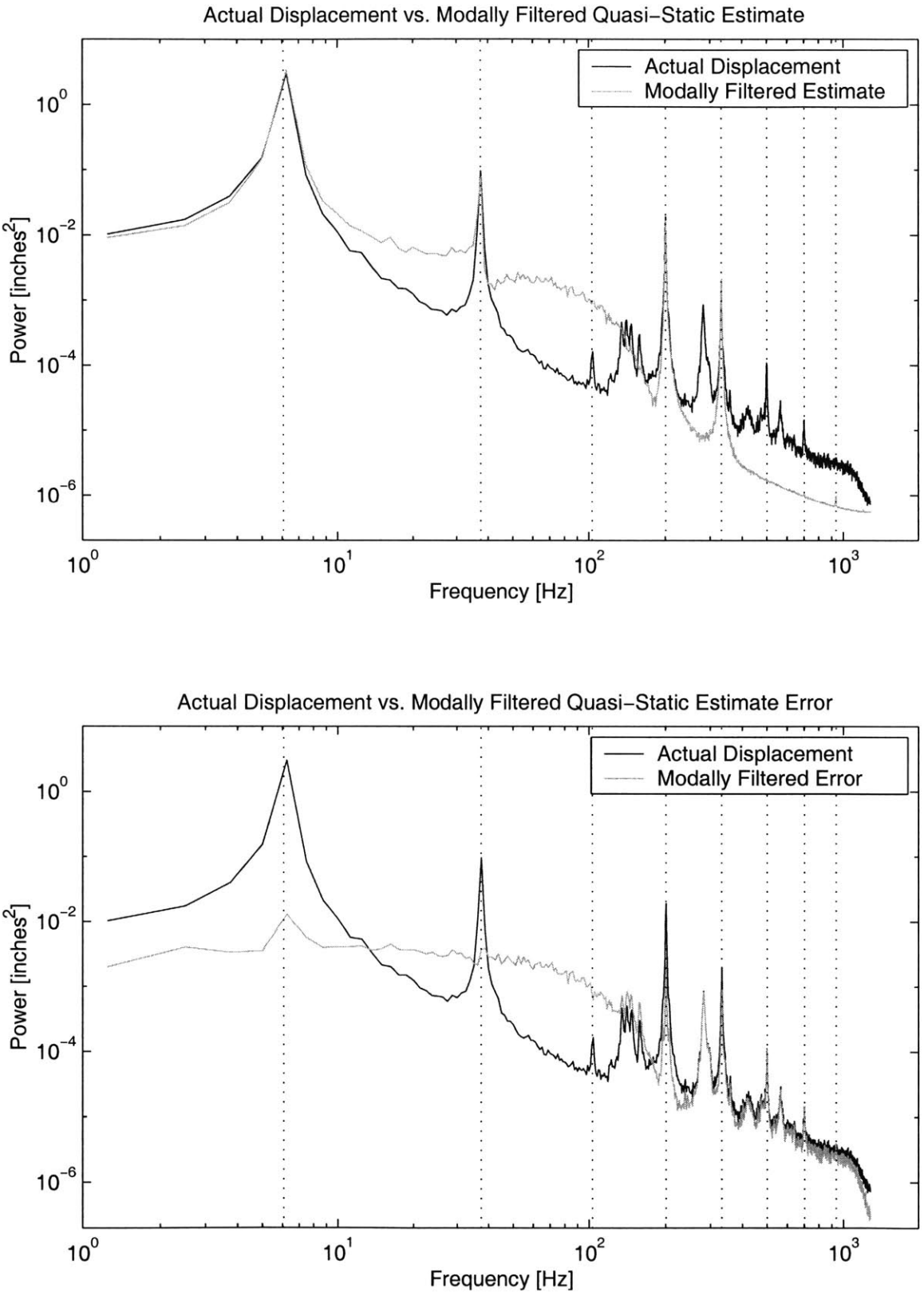


Figure 4-19: Power Spectral Density of the Modally Filtered Quasi-Static Estimate vs. Actual Displacement

Above the fifth mode, the 8 Mode Kalman Filter still tries to make an estimate of the position, even though it is largely unsuccessful. One of the reasons is that the noise floor on the laser displacement sensor is higher than the levels the Kalman filter is trying to estimate.

Since all four methods estimate the individual modes directly it is helpful to compare the estimates on a mode by mode basis. Especially telling is the first mode shown in Figure 4-22. At low frequency the Kalman filter produces a poorer estimate than the other three. At the peak of the first mode all four methods produce nearly the same results. The Kalman Filter's response then rolls-off as $1/\omega^2$ as would be expected of a second order pole.

The results of the other three methods continue on for a while until the MFQS method is able to roll-off. A higher order filter would be able to produce a more immediate roll-off but would not be implementable in real-time. At the fifth mode the Filtered case rolls-off, it rolls off faster than the $1/\omega^2$ caused by the amplifier dynamics. The Quasi-Static method seems to roll-off but that is an artifact of the amplifier dynamics.

After the fifth mode the two filtering methods have no interesting dynamics since they have been filtered out. The Quasi-Static method produces peaks at the sixth mode and higher which is the result of aliasing. The Kalman filter also has a peak at the seventh mode which is most likely due to modeling error.

Figures 4-23– 4-26 show the comparisons for the next four modes of the structure. Each of the plots show similar trends as the plot for the first mode. The Kalman Filter continues to show a behavior indicative of a second order pole for each mode. The Quasi-Static case has aliasing at high frequencies that is more or less accounted for with the filtering.

Figure 4-24 is worth mentioning separately however. The third mode has very low magnitude overall due to the position of the piezo-ceramic. This means that the estimated transfer function was noisier than the other methods resulting in greater errors. This is reflected, in the figure, by the peaks at the neighboring modes.

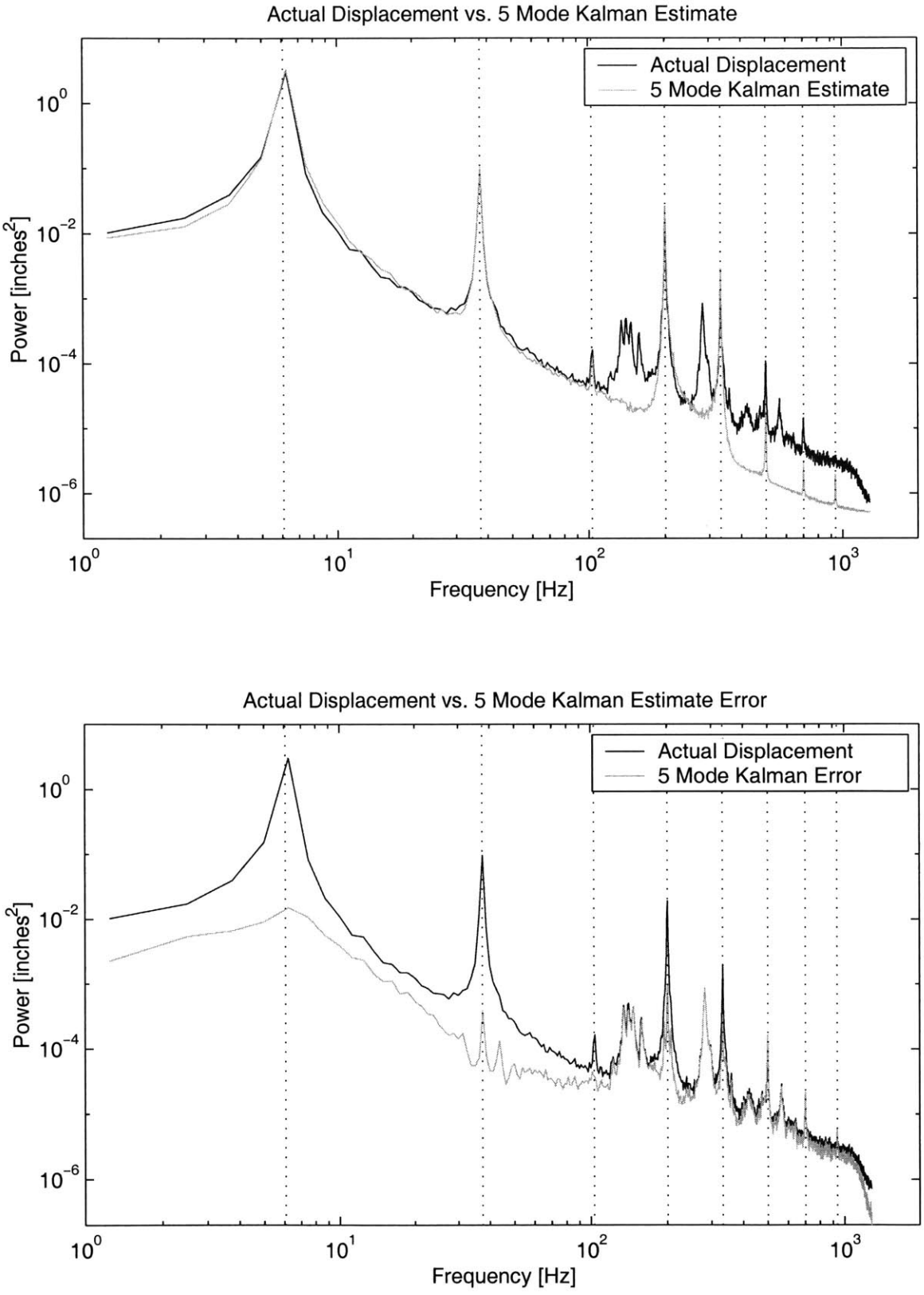


Figure 4-20: Power Spectral Density of the 5 Mode Kalman Filter Estimate vs. Actual Displacement

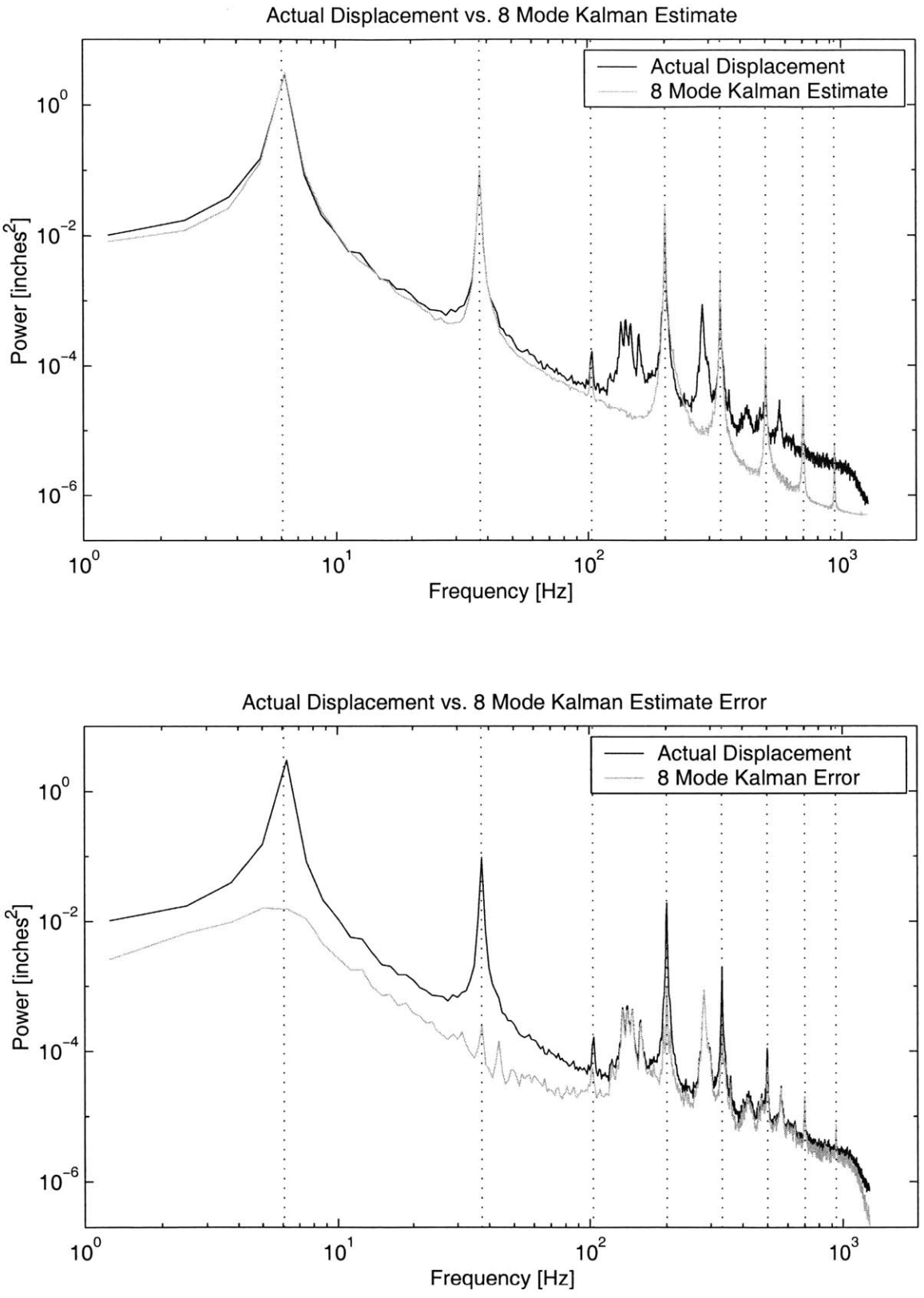


Figure 4-21: Power Spectral Density of the 8 Mode Kalman Filter Estimate vs. Actual Displacement

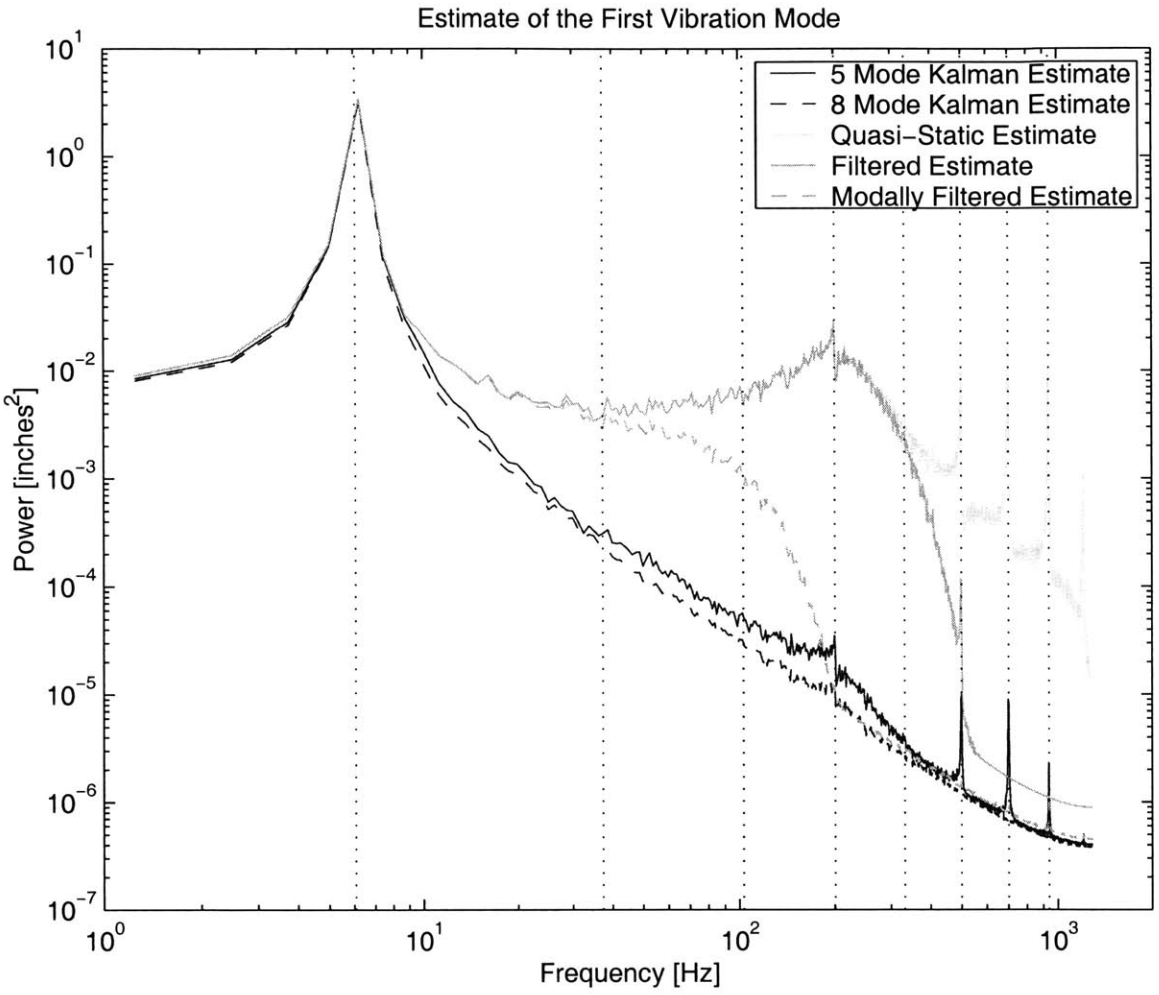


Figure 4-22: Power Spectral Density of Estimates for the First Mode

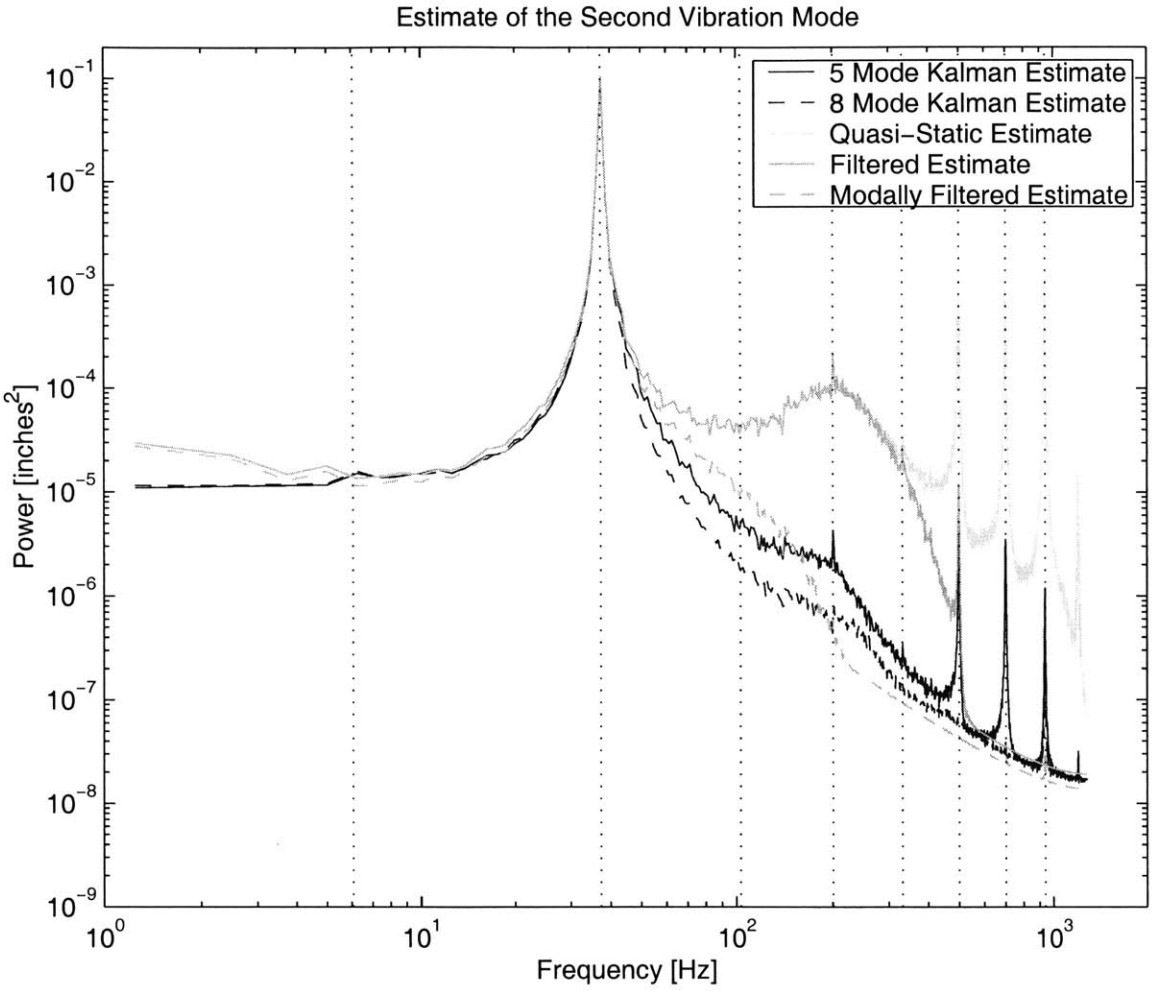


Figure 4-23: Power Spectral Density of Estimates for the Second Mode

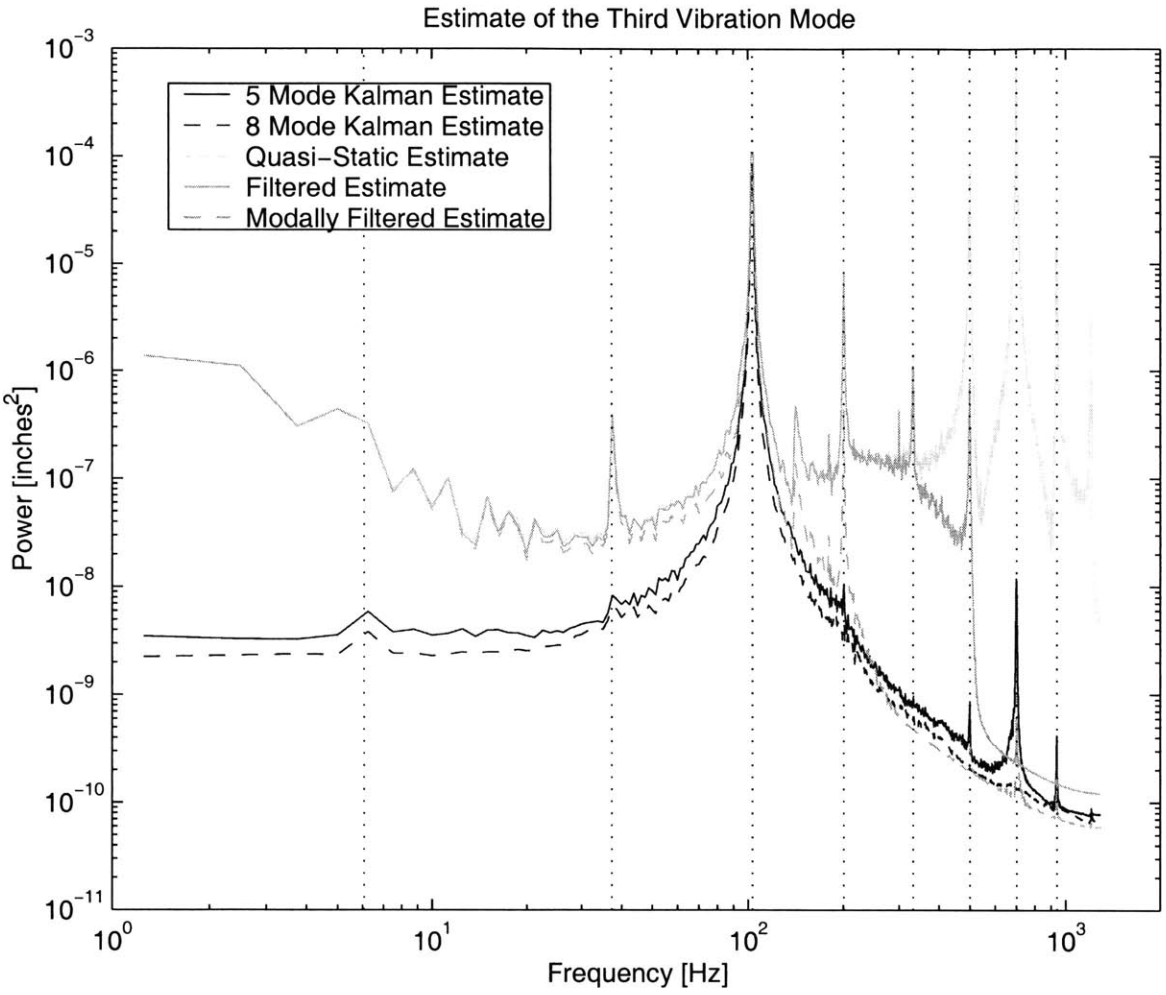


Figure 4-24: Power Spectral Density of Estimates for the Third Mode

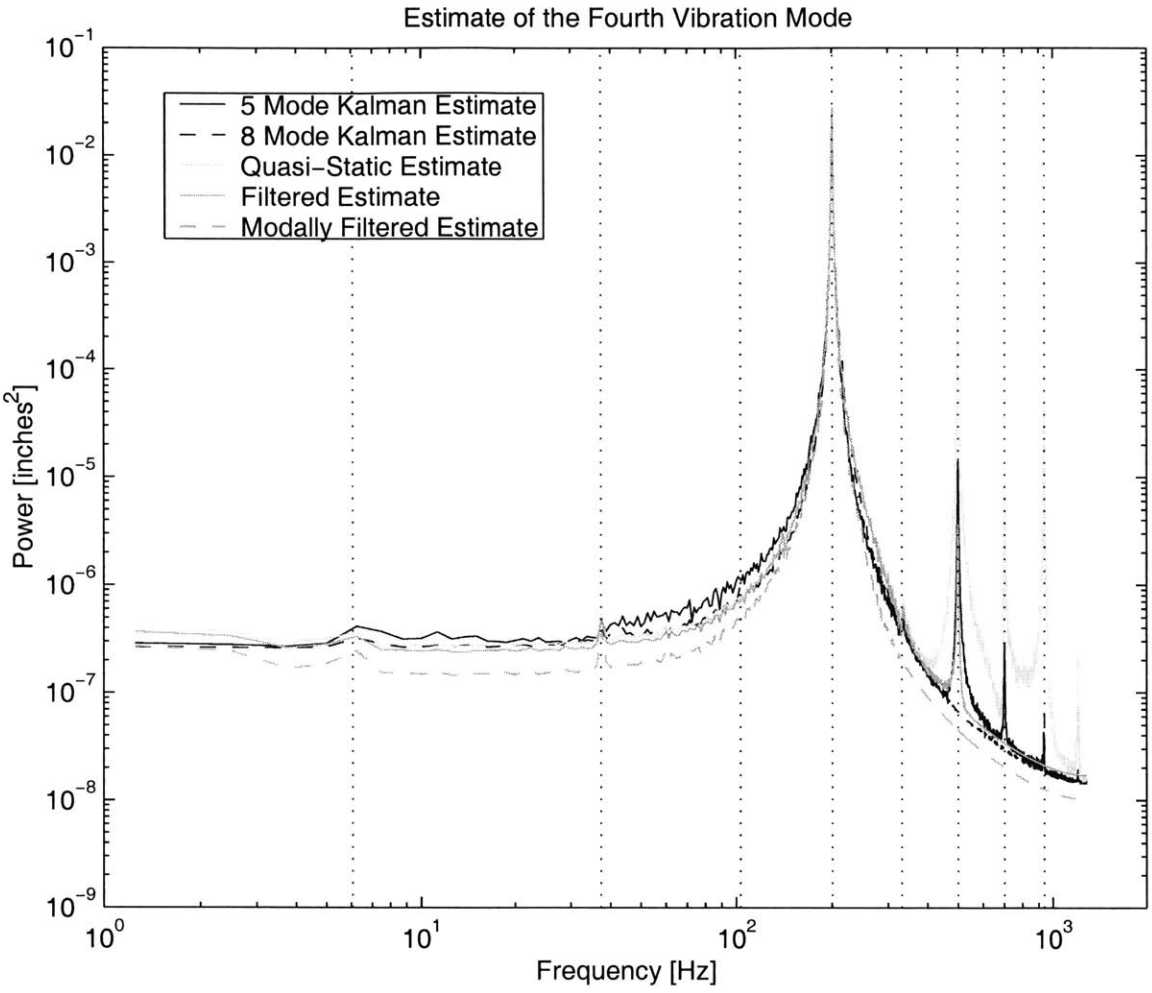


Figure 4-25: Power Spectral Density of Estimates for the Fourth Mode

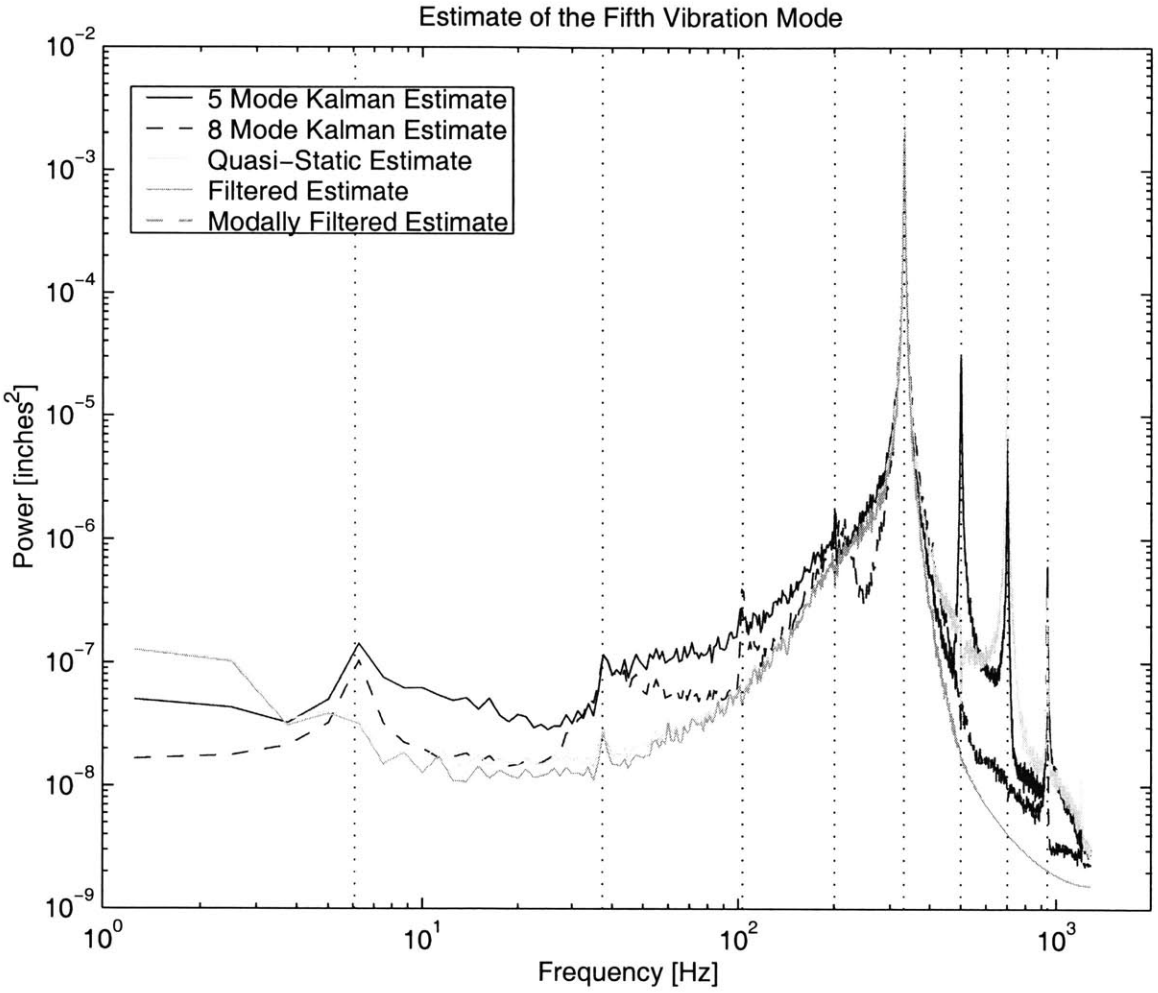


Figure 4-26: Power Spectral Density of Estimates for the Fifth Mode

4.5 Summary

The experiment shows results very similar to those in the simulation, particularly the simulations that included both modeling and sensor errors. The Kalman Filter was able to best predict the deflection of the tip of the beam. The Modally Filter Quasi-Static estimate matched the plain Filtered Quasi-Static method presented in the simulations.

The effects of the Filtered Quasi-Static method did not prove as profound as they were in the simulations. One of the main reasons for this is that the amplifier dynamics tremendously decreased the effects of aliasing by not exciting the higher modes. However, the power spectral density plots, Figures 4-17 and 4-18, clearly show that the high frequency modes that result in errors for the Quasi-Static method, are filtered correctly.

Chapter 5

Conclusions

5.1 Summary

The purpose of this paper is to explore the use of Dynamic Methods for Shape Estimation, specifically to apply temporal filters to static methods in an attempt to improve their usefulness.

Both from the simulation data and from the experimental results it is clear that the Quasi-Static shape estimation method is a poor choice for Dynamic Structural Shape Estimation. The aliasing effects of the higher modes is simply too great.

The reason that static methods are able to achieve such good results for a static structure is that the complexity of the deformation is limited. To disturb a higher mode statically requires complex loadings which vary within small lengths. This corresponds to the shorter structural wavelengths of the higher modes. In a dynamic structure the higher modes can be actuated by high frequency disturbances, rather than requiring complicated structural variations in disturbance.

As an answer to the problem of aliasing, control type methods can be utilized for shape estimation. The controls methods aid in the discernment of the modes in terms of frequency, while the shape estimation methods sort the modes spatially. The shape estimation methods also provide the basis for turning the modal amplitudes into global deformation estimates.

Adding a simple filter to the Quasi-Static method resulted in the redction of error

by an order of magnitude in the simulations. The experimental results are not as good, however. The reason for this is largely because the amplifier's dynamics provided a natural filter for the system. A broader spectrum white noise disturbance would most likely show a more marked improvement when using the filter.

In the simulations the filtering of the Quasi-Static method provided ample reduction of error so the addition of the multiple filters was not investigated. In the experiment this was not the case so a filter for each sensed mode was used. Using a separate filter for each mode dramatically decreases the total error.

The Kalman Filter was the best filter overall. In both the simulations and the experiment the Kalman filter was able to reduce the error tremendously. The results of the 5 Mode Kalman Filter as compared to the 8 Mode Kalman Filter, used in the experiments, paralleled the results of the simulations. For high levels of noise the additional modes that are modeled do not appreciably decrease the error.

Higher order methods are not always needed however. When the structure has a low bandwidth disturbance, all of the methods provide roughly the same level of performance. The main difference is that a Kalman Filter requires additional computation, and the Filtered Quasi-Static method can introduce unnecessary lags and extra components. The simplest method that meets the requirements should be used.

5.2 Future of Shape Estimation

Shape estimation for use in real structures is becoming a possibility. Advances in the speed of processors and the ability to interrogate more sensors at one time enable larger arrays. As the size of the arrays increase and the accuracy of modeling methods increase the error in the estimated deformation decreases dramatically as shown here.

The ideal method needs to be flexible enough to handle normal changes in the structure. It should also be able to be adjusted on a regular basis. Since if a structure is damaged, the result can severely and adversely effect the accuracy of the method.

Some of the model updating can occur during regular operation. To do this the

Fourier transforms of the sensors can be compared. When a drift from a consistent ratio is found the model can be adjusted. The model can also be updated by using a laser vibrometer at predetermined calibration points, when the structure is off-line.

The key to successful use of Shape Estimation is to not oversimplify. Adding a simple temporal filter to a shape estimation method can reduce the error by an order of magnitude. Adding a more sophisticated method can improve the performance even further.

Appendix A

Additional Simulation Results

Note: All values are as the RMS error as a percentage of the RMS deflection.

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	104.74	67.02	27.69	16.34	10.74	7.02
1 %	110.65	63.70	29.63	16.80	11.26	7.26
5 %	116.22	67.82	30.12	17.52	11.47	7.53
10 %	112.50	67.92	29.92	17.03	11.39	7.71
20 %	105.07	67.79	28.71	17.50	12.06	8.57

Table A.1: Quasi-Static Method - Accurate Model, No Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	99.16	71.07	31.25	17.94	11.55	7.48
1 %	122.20	68.45	28.73	17.19	11.06	7.20
5 %	107.31	62.66	28.40	16.29	10.51	6.91
10 %	118.94	71.85	31.45	17.52	11.38	7.88
20 %	109.13	74.17	29.83	18.07	12.12	8.76

Table A.2: Quasi-Static Method - With 5% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	119.27	69.85	31.12	16.84	11.28	7.53
1 %	112.41	65.70	30.70	17.06	10.96	7.35
5 %	121.10	65.04	29.06	16.66	10.89	7.25
10 %	113.89	73.66	31.35	18.30	11.99	8.14
20 %	107.25	71.82	30.11	17.98	12.18	8.79

Table A.3: Quasi-Static Method - With 20% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	11.88	2.97	0.68	0.24	0.15	0.08
1 %	12.80	2.97	0.69	0.27	0.18	0.14
5 %	13.43	3.31	0.76	0.52	0.54	0.59
10 %	13.35	3.14	0.99	0.89	1.02	1.13
20 %	11.88	3.10	1.50	1.70	1.96	2.20

Table A.4: Quasi-Static Method With Filtering - Accurate Model, No Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	11.13	3.54	0.70	0.27	0.16	0.08
1 %	14.65	3.06	0.68	0.26	0.18	0.14
5 %	12.60	3.22	0.73	0.48	0.50	0.54
10 %	13.26	5.80	0.86	0.28	0.13	0.07
20 %	12.13	5.37	0.86	0.30	0.18	0.13

Table A.5: Quasi-Static Method With Filtering - With 5% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	14.13	3.25	0.69	0.27	0.17	0.10
1 %	12.52	3.29	0.67	0.30	0.18	0.15
5 %	14.10	3.21	0.72	0.49	0.52	0.56
10 %	13.67	5.72	0.90	0.27	0.13	0.07
20 %	13.23	5.94	0.89	0.31	0.18	0.14

Table A.6: Quasi-Static Method With Filtering - With 20% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	36.0	6.36	1.20	0.44	0.21	0.12
1 %	39.2	6.07	1.23	0.46	0.22	0.12
5 %	40.9	6.60	1.25	0.50	0.26	0.17
10 %	40.3	6.45	1.30	0.54	0.34	0.26
20 %	36.0	6.40	1.36	0.72	0.54	0.47

Table A.7: Exact Kalman Filter - Accurate Model, No Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	84.03	14.05	1.82	0.57	0.26	0.14
1 %	109.52	11.97	1.69	0.54	0.25	0.13
5 %	94.72	12.34	1.66	0.54	0.26	0.17
10 %	99.16	71.07	31.25	17.94	11.55	7.48
20 %	122.20	68.45	28.73	17.19	11.06	7.20

Table A.8: Exact Kalman Filter - With 5% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	104.17	12.65	1.72	0.56	0.26	0.14
1 %	92.90	12.85	1.71	0.56	0.25	0.14
5 %	108.16	12.35	1.62	0.55	0.27	0.17
10 %	96.41	13.72	1.90	0.67	0.39	0.29
20 %	91.03	13.37	1.85	0.82	0.58	0.50

Table A.9: Exact Kalman Filter - With 20% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	9.12	2.01	0.47	0.19	0.094	0.059
1 %	9.14	2.10	0.49	0.20	0.10	0.069
5 %	9.44	2.18	0.53	0.25	0.17	0.14
10 %	8.87	2.12	0.60	0.35	0.28	0.24
20 %	9.14	2.16	0.82	0.61	0.52	0.47

Table A.10: Extended Kalman Filter - Accurate Model, No Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	12.66	5.03	0.85	0.27	0.12	0.06
1 %	11.97	4.74	0.80	0.26	0.12	0.07
5 %	11.20	4.56	0.78	0.28	0.17	0.13
10 %	12.20	4.61	0.94	0.42	0.31	0.26
20 %	11.70	4.88	1.07	0.66	0.54	0.49

Table A.11: Extended Kalman Filter - With 5% Model Error

Sensor Noise	Number of Sensors					
	1	2	4	6	8	10
0 %	11.86	4.94	0.81	0.26	0.12	0.06
1 %	12.24	4.82	0.81	0.26	0.12	0.06
5 %	11.76	4.62	0.79	0.29	0.17	0.13
10 %	13.06	5.03	0.95	0.43	0.31	0.27
20 %	12.09	5.02	1.13	0.68	0.55	0.49

Table A.12: Extended Kalman Filter - With 20% Model Error

Appendix B

Conversion of Continuous State Space Model to Discrete Form

In modal form the structure is modeled:

$$\ddot{\eta}_n + 2\zeta_n\omega_n\dot{\eta}_n + \omega_n^2\eta_n = F_n \tag{B.1}$$

Form x such that:

$$x = \begin{bmatrix} \dot{\eta} \\ \eta \end{bmatrix} \tag{B.2}$$

The Continuous State Space Form:

$$\dot{x} = Ax + B_w wy = Cx + D_w w \tag{B.3}$$

$$A = \left[\begin{array}{c|c} 0 & I \\ \hline -2\zeta_n\omega_n & -\omega_n^2 \end{array}\right] \tag{B.4}$$

$$B = \left[\begin{array}{c} 0 \\ \hline F_n \end{array}\right] \tag{B.5}$$

For the Discrete State Space Model:

$$x_{k+1} = \Phi x_k + w_k \tag{B.6}$$

$$z_k = H_k x_k + v_k \tag{B.7}$$

For the continuous form, z is completely analogous to y for the discrete form, therefore:

$$H_k = C \tag{B.8}$$

for time invariant systems

The w_k and v_k are the disturbance matrix and the sensor noise matrix respectively, where their error covariances are given by:

$$E\left[w_k w_i^T\right] = \left\{ \begin{array}{ll} Q_k & i = k \\ 0 & i \neq k \end{array} \right\} \tag{B.9}$$

$$E\left[v_k v_i^T\right] = \left\{ \begin{array}{ll} R_k & i = k \\ 0 & i \neq k \end{array} \right\} \tag{B.10}$$

The state transition matrix, Φ is calculated from the A matrix:

$$\Phi = \exp(A \times \Delta t) \tag{B.11}$$

where *exp* is the matrix exponential function

Finally create Ξ (W is the power spectral density matrix associated with the forcing):

$$\Xi = \left[\begin{array}{c|c} -A & BWB^T \\ \hline 0 & A^T \end{array} \right] \tag{B.12}$$

$$G = \exp(\Xi) = \left[\begin{array}{c|c} \cdots & \Phi^{-1} Q_k \\ \hline 0 & \Phi^T \end{array} \right] \tag{B.13}$$

The upper left partition of G is not important. Transposing the lower right par-

tition gives Φ. To obtain Q_K:

$$Q_k = \Phi * (\textit{upper right partition of } G) \tag{B.14}$$

Appendix C

Matlab Simulation Code

C.1 Finite Element Model

The following is the code that was used to perform the Finite Element Analysis

---

```
at10.0pt% This is a finite element code for a beam
% If the model level is set to 1 it includes the effect of a piezoceramic
% The dimensions are identical to those in the experiment

FF=1;
N=240*FF;
model=0;  %toggles the FE to model mode.
modes=max(25,sensors*5);
L=.381; %1.00;
l=L/N;                                                          10
b=3.81e-2;
t=1.016e-3; %7.62e-4;
h=t;
h2=2.54e-4;
E=210e9; %70.33e9;
E2=66e9*model;
rho=7850; %2700;
rho2=7500*model;
Ey=E2/E;
```

```
I=b*h^3/12;                                                                    20
I2=b/12*(h^4+13*Ey*h2*h^3+12*Ey*h2^2*h^2+4*Ey*h2^3*h+Ey^2*h2^4)/(Ey*h2+h)*1;
m=b*h*rho;  % mass per length of beam
m2=m+h2*b*rho2;

if model==0
switch sensors
  case 1
    gageloc=round([3]*FF);
  case 2
    gageloc=round([3 120]*FF);                                                 30
  case 4
    gageloc=round([3 60 120 180]*FF);
  case 6
    gageloc=round([3 40 80 120 160 200]*FF);
  case 8
    gageloc=round([3 30 60 90 120 150 180 210]*FF);
  case 10
    gageloc=round([3 24 48 72 96 120 144 168 192 216]*FF);
  otherwise
    gageloc=round([3 48 96 144 192]*FF);                                       40
end
else
  gageloc=round([3 48 96 144 192]*FF);
  sensors=5;
end

baseK=[12 6*l -12 6*l;
6*l 4*l^2 -6*l 2*l^2;
-12 -6*l 12 -6*l;
6*l 2*l^2 -6*l 4*l^2]*E*I/l^3;                                                 50

baseM=[156 22*l 54 -13*l;
22*l 4*l^2 13*l -3*l^2;
54 13*l 156 -22*l;
-13*l -3*l^2 -22*l 4*l^2]*m*l/420;
```

```
baseK2=[12 6*l −12 6*l;
6*l 4*l^2 −6*l 2*l^2;
−12 −6*l 12 −6*l;
6*l 2*l^2 −6*l 4*l^2]*E*I2/l^3;

baseM2=[156 22*l 54 −13*l;
22*l 4*l^2 13*l −3*l^2;
54 13*l 156 −22*l;
−13*l −3*l^2 −22*l 4*l^2]*m2*l/420;

K=zeros(N*2+2);
M=zeros(N*2+2);

for z=1:N
  scale(2*z)=z*l;
  scale(2*z−1)=sqrt(−1)*z*l;
  scale2(z)=z*l;
  for i=1:4
    for j=1:4
      if (z>4*FF)&(z<=44*FF)
        K((z−1)*2+i,(z−1)*2+j)=K((z−1)*2+i,(z−1)*2+j)+baseK2(i,j);
        M((z−1)*2+i,(z−1)*2+j)=M((z−1)*2+i,(z−1)*2+j)+baseM2(i,j);
      else
        K((z−1)*2+i,(z−1)*2+j)=K((z−1)*2+i,(z−1)*2+j)+baseK(i,j);
        M((z−1)*2+i,(z−1)*2+j)=M((z−1)*2+i,(z−1)*2+j)+baseM(i,j);
      end
    end
  end
end

clear K2
clear M2

K2=sparse(K(3:N*2+2,3:N*2+2));
M2=sparse(M(3:N*2+2,3:N*2+2));
```

```
options.tol=1e-12;
options.disp=0;
[V,Do]=eigs(K2,M2,modes at10.0pt,'sm',options);

Dd=diag(Do);

[D2,II]=sort(abs(Dd));
                                                                    100
omega=sqrt(D2(1:modes));

%sorts the modes by eigenvalues
clear V2
for z=1:length(II)
  V2(:,z)=V(:,II(z));
end

%%%%%%%%
                                                                    110
Mr=(V2'*M2*V2);
Kr=(V2'*K2*V2);

V3=real(V2/sqrt(Mr));  %Mass normalized eigenfunctions

% A nice thing about mass normalized eigenfunctions is that with
% unity magnitude they all have a 1 unit tip deflection

for z=1:length(V2)
  if mod(z,2)==1                                                    120
    disp((z+1)/2,:)=V3(z,:);
  else
    slope(z/2,:)=V3(z,:);
  end
end

for z=1:modes
```

```
 if disp(N,z)<0
   disp(:,z)=-disp(:,z);
   slope(:,z)=-slope(:,z);                                          130
 end
end

%%%%  Readings for each strain sensor for first 'modes' modes

for ii=1:modes
 for jj=1:sensors
%    if jj==1
%      curv(jj,ii)=slope(1,ii)/l;
%    else                                                           140
     curv(jj,ii)=(slope(gageloc(jj)+2*FF,ii)-slope(gageloc(jj)-2*FF,ii))/2/l;
%    end
 end
end

sensmat=inv(curv(:,1:sensors));

zeta=.002;

A=[zeros(modes) eye(modes);                                         150
  -diag(omega(1:modes).^2) -diag(omega(1:modes)*zeta*2)];

if model==0
 B=[zeros(modes,1);
 disp(240,:)'];  % tip forcing
else
 B=[zeros(modes,1);
  (V3(4*2,1:modes)-V3(44*2,1:modes))']; %piezo forcing (as in experiment)
end
                                                                    160
Cs=[sensmat*curv zeros(sensors,modes)];          % Actual reading of sensor

D=0;
```

```
Creal=[curv zeros(sensors,modes)]; %output is strain readings
CsC=[eye(modes) zeros(modes)]; % Output is modal amplitudes

syss=ss(A,B,Cs,D);
syssC=ss(A,B,CsC,D);
sysreal=ss(A,B,Creal,D);                                          170
```

C.2 Simulation Code

This code was used to run the simulations of the Kalman Filter, and Quasi-Static Estimator.

```
% This matlab code performs simulations of the Kalman filter and the other
% shape estimation methods

switch sensors
  case 1
    efg=3.1;efg2=3.4;
  case 2
    efg=2.6;efg2=2.9;
  case 4
    efg=2.0;efg2=2.5;                                             10
  case 6
    efg=1.75;efg2=2.2;
  case 8
    efg=1.5;efg2=1.875;
  case 10
    efg=1.3333;efg2=1.5;
end
kmodes=max(sensors*2,5);
twice=1;
if floor(ef/20)==1                                                20
ploton=0;
else
ploton=0;
```

```
end
cumplot=floor(ef/20)*0;

kmodes2=sensors;
amps=abs(B(modes+1:modes*2)./omega.^2);
amps=amps/max(amps);

baseXo=zeros(modes,1);
baseXo=8e-4.*amps;
Xo=baseXo.*rand(size(baseXo));
baseXodot=baseXo.*omega;
Xodot=baseXodot.*rand(size(baseXodot));
Xotot=[Xo Xodot];

deltat=2/2^15;
freq=omega/2/pi;
time=0:deltat:deltat*2^15-deltat;
u=makeu(sqrt(omega(10)*omega(11))*0,time);  %input forcing
clear Z X
fprintf('Running Simulation of Vibrationsr')

[Xs,T]=lsim(syssC,u,time,Xotot);  %Modal Amplitudes
Xs=Xs(2^14+1:2^15,:);
time=time(1:2^14);
freqs=[linspace(0,1/deltat/2,length(time)/2) linspace(-1/deltat/2,0,length(time)/2)];
Z=Xs*curv'+(rand(length(Xs),sensors)-.5)*.4*noise;
Zact=Xs*curv';
X=Xs;

Am=[A(1:kmodes,1:kmodes) A(1:kmodes,modes+1:modes+kmodes)
  A(modes+1:modes+kmodes,1:kmodes) A(modes+1:modes+kmodes,modes+1:modes+kmodes)];

%fprintf('Running Simulation of Vibrations - Done!');
BB=[B(1:kmodes); B(modes+1:modes+kmodes)];
QQ=BB*BB';
AA=[-Am QQ; zeros(size(Am)) Am']*deltat;
```

```
BC=expm(AA);                                                              60
Phi=expm(Am*deltat);
Q=Phi*BC(1:kmodes*2,kmodes*2+1:kmodes*4);
Q=(Q+Q')/2;
P=eye(kmodes*2)*1e-3*0;zeros(10,10);

Xhat=zeros(length(time),kmodes);
Xhatdot=zeros(length(time),kmodes);
H=[curv(1:sensors,1:kmodes) zeros(sensors,kmodes)];
Rbase=eye(sensors)*mean(std(Z));
R=Rbase*10^(efg);                                                         70

Xhat(1,:)=X(1,1:kmodes);
Xhatdot(1,:)=(X(2,1:kmodes)-X(1,1:kmodes))/deltat;

if twice
  Am2=[A(1:kmodes2,1:kmodes2) A(1:kmodes2,modes+1:modes+kmodes2)
  A(modes+1:modes+kmodes2,1:kmodes2) A(modes+1:modes+kmodes2,modes+1:modes+kmodes2)];
  BB2=[B(1:kmodes2); B(modes+1:modes+kmodes2)];
  QQ2=BB2*BB2';
  AA2=[-Am2 QQ2; zeros(size(Am2)) Am2']*deltat;                           80
  BC2=expm(AA2);
  Phi2=expm(Am2*deltat);
  Q2=Phi2*BC2(1:kmodes2*2,kmodes2*2+1:kmodes2*4);
  Q2=(Q2+Q2')/2;
  P2=eye(kmodes2*2)*1e-3*0;
  Xhat2=zeros(length(time),kmodes2);
  Xhatdot2=zeros(length(time),kmodes2);
  H2=[curv(1:sensors,1:kmodes2) zeros(sensors,kmodes2)];
  Rbase2=eye(sensors)*mean(std(Z));
  R2=Rbase2*10^(efg2);                                                    90
  Xhat2(1,:)=X(1,1:kmodes2);
  Xhatdot2(1,:)=(X(2,1:kmodes2)-X(1,1:kmodes2))/deltat;
end

Out=0;
```

```
fprintf('Running Kalman Filter                r');
if 1==1
for ii=1:3000
 if sign(diag(P))>=0                                                    100
  Pm=Phi*P*Phi'+Q;
  P=(eye(kmodes*2)-Pm*H'*inv(H*Pm*H'+R)*H)*Pm;
  if twice
    Pm2=Phi2*P2*Phi2'+Q2;
    P2=(eye(kmodes2*2)-Pm2*H2'*inv(H2*Pm2*H2'+R2)*H2)*Pm2;
  end
 else
  break;
 end
end                                                                     110
end
K=Pm*H'*inv(H*Pm*H'+R);

if twice
  K2=Pm2*H2'*inv(H2*Pm2*H2'+R2);
end

for T=2:length(time)
   if mod(T,2^10)==0
     T/length(time)*100;                                               120
   end
   if sign(diag(P))>=0
    TT=Phi*[Xhat(T-1,:) Xhatdot(T-1,:)]';
%    Pm=Phi*P*Phi'+Q;  %These lines can be reincluded if there is initial conditions
    Xhatm=TT(1:kmodes);
    Xhatdotm=TT(kmodes+1:kmodes*2);
%    K=Pm*H'*inv(H*Pm*H'+R);
    TTB=TT+K*(Z(T,:)'-H*TT);
    Xhat(T,:)=TTB(1:kmodes)';
    Xhatdot(T,:)=TTB(kmodes+1:kmodes*2)';                              130
%    P=(eye(kmodes*2)-K*H)*Pm;
```

```
   if twice
     TT2=Phi2*[Xhat2(T-1,:) Xhatdot2(T-1,:)]';
%      Pm2=Phi2*P2*Phi2'+Q2;
     Xhatm2=TT2(1:kmodes2);
     Xhatdotm2=TT2(kmodes2+1:kmodes2*2);
%      K2=Pm2*H2'*inv(H2*Pm2*H2'+R2);
     TTB2=TT2+K2*(Z(T,:)'-H2*TT2);
     Xhat2(T,:)=TTB2(1:kmodes2)';
     Xhatdot2(T,:)=TTB2(kmodes2+1:kmodes2*2)';                    140
%      P2=(eye(kmodes2*2)-K2*H2)*Pm2;
   end
  else
   Out=1;
  end
  if Out==1
   break
  end
end
                                                                  150
W=[sqrt(omega(sensors)*omega(sensors+1))/2/pi];
zz=fft(Z);
nyquis=1/deltat/2;htz=W;
tmax=max(time)+deltat;
zz(round(W*tmax)+1:length(time)-round(W*tmax),:)=0;

Zf=real(ifft(zz));%filter2(BzF,Z')';%

Xstat=(inv(curv(1:sensors,1:sensors))*Z')';
Xf=(inv(curv(1:sensors,1:sensors))*Zf')';                         160

dispact=X(:,1:modes)*disp(240,1:modes)';
dispact5=X(:,1:sensors)*disp(240,1:sensors)';  %doesn't matter much
dispest=Xhat(:,1:kmodes)*disp(240,1:kmodes)';
dispest5=Xhat(:,1:sensors)*disp(240,1:sensors)';  %Probably no good.
dispst=Xstat(:,1:sensors)*disp(240,1:sensors)';
dispstf=Xf(:,1:sensors)*disp(240,1:sensors)';
```

```
cumact(:,ef)=dispact;
cumest(:,ef)=dispest;                                                    170
cumst(:,ef)=dispst;
cumstf(:,ef)=dispstf;

if twice
 dispest2=Xhat2(:,1:kmodes2)*disp(240,1:kmodes2)';
% errest2(ef,ff)=std(dispest2-dispact);
 cumest2(:,ef)=dispest2;
end

if ploton                                                                180
figure(1)
clf
subplot(2,1,1)
plot(time,dispact)
hold on
plot(time,dispest,'r')
plot(time,dispest2,'m')
plot(time,dispst,'g')
plot(time,dispact)
plot(time,dispest,'r')                                                   190
plot(time,dispest2,'m')
subplot(2,1,2)
plot(time,dispest-dispact,'r')
hold on
plot(time,dispest2-dispact,'m')
plot(time,dispstf-dispact,'b')

figure(2)
clf
plot(time,dispact-dispest)                                               200
hold on
plot(time,dispact-dispact5,'r')
plot(time,dispact-dispst,'g')
```

```
plot(time,dispact-dispest5,'m')

figure(3)
clf
loglog(fftshift(freqs),fftshift(abs(fft(dispact-dispest))))
hold on
semilogy(fftshift(freqs),fftshift(abs(fft(dispact-dispst))),'g')
semilogy(fftshift(freqs),fftshift(abs(fft(dispact-dispest5))),'m')
semilogy(fftshift(freqs),fftshift(abs(fft(dispact-dispest))))
semilogy(fftshift(freqs),fftshift(abs(fft(dispact-dispstf))),'r')
if twice
  semilogy(fftshift(freqs),fftshift(abs(fft(dispact-dispest2))),'c')
end
loglog(fftshift(freqs),fftshift(abs(fft(dispact))),'k')
axx=axis;
axis([0 axx(2:4)]);
for ii=1:modes
  if freq(ii)<=max(freqs)
    plot([freq(ii),freq(ii)],axx(3:4),'k:');
  end
end
end

if cumplot
dif=1;
figure(5)
clf
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumact))')),'r')
hold on
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumest-cumact*dif))')),'b')
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumest2-cumact*dif))')),'g')
axx=axis;
axis([0 axx(2:4)]);
for ii=1:modes
  if freq(ii)<=max(freqs)
    plot([freq(ii),freq(ii)],axx(3:4),'k:');
```

```
 end                                                                    240
end

figure(6)
clf
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumact))')),'r')
hold on
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumst-cumact*dif))')),'b')
loglog(fftshift(freqs),fftshift(mean(abs(fft(cumstf-cumact*dif))')),'g')
axx=axis;
axis([0 axx(2:4)]);                                                     250
for ii=1:modes
 if freq(ii)<=max(freqs)
  plot([freq(ii),freq(ii)],axx(3:4),'k:');
 end
end
end
```

C.3 Experiment Analysis Code

This code was used to analyze the data from the experiments:

```
% Code to analyze the result of the experiment

load tf      % The
load r2000.txt  %This is the time response gathered in the experiment
temp=r2000;
omega=omega/2/pi;
L=length(omega)-1;
LL=length(temp);
bh=1/temp(LL,1);
used=8;                                                                 10
time=max(temp(:,1));
Om(1)=0;   %Om is the frequencies for the fft
for jj=1:LL/2
 Om(jj+1)=(jj)/time;
```

```
  Om(LL+1-jj)=-Om(jj+1);
end
Omf=fftshift(Om);  % use this with the fftshift of the fft for more clarity

expkalmimproved;  % Kalman Filter code
                                                                    20
H=[CE(2:6,1:used) zeros(5,used)];
hh=inv(H(1:5,1:5));
Xs=zeros(5,LL);
Zt=Z';
clear QB
Xs=hh*Z;
W=[sqrt(omega(5)*omega(6))]*time/length(Z)*2;
if W<1
  QB=fir1(N,W);
else                                                                30
  QB=1;
end

%Quasi Static Method

z=fft(Z')';
W2=[sqrt(omega(5)*omega(6))];

%Quasi Filtered
                                                                    40
tmax=max(time)+deltat;
if W2*tmax<length(z)/2
  z(:,round(W2*tmax)+1:length(t)-round(W2*tmax))=0;
end
Xf=hh*filter2(QB,Z);
Xf2=hh*real(ifft(z')');

stXs=std(disp-sum(Xs));
maxXs=max(abs(disp-sum(Xs)));
errst=disp'-sum(Xs)'-mean(disp'-sum(Xs)');                          50
```

```
dispK=sum(Xhat);
dispXs=sum(Xs);
dispXf=sum(Xf);
dispXf2=sum(Xf2);
ref=std(disp(1000:7000));

errK=std(dispK(1000:7000)-disp(1000:7000))/ref
errXs=std(dispXs(1000:7000)-disp(1000:7000))/ref
errXf=std(dispXf(1000:7000)-disp(1000:7000))/ref                    60
errXf2=std(dispXf2(1000:7000)-disp(1000:7000))/ref

Lt=length(t)
freq=1/t(2)/Lt;

for jj=1:Lt/2
  freq(jj)=freq(1)*jj;
  freq(Lt-jj+1)=-freq(jj);
end
clf                                                                 70
loglog(freq,abs(fft(disp-dispK)))
hold on
loglog(freq,abs(fft(disp)),'r')

ploton=0;
if ploton==1
  figure(1)
  clf
  plot(disp,'c');
  hold on                                                           80
  plot(Xhat(1,:),'r')
  plot(Y1)
  plot(Xs(1,:),'m');

  figure(1)
  clf
```

```
plot(disp-dispX,'r');
hold on
plot(disp-sum(Xs));
legend('Kalman Error','Static Error');

figure(2)
clf
loglog(Omf,fftshift(abs(fft(disp))));
hold on
plot(Omf,fftshift(abs(fft(dispX))),'r');
plot(Omf,fftshift(abs(fft(sum(Xs)))),'g');
semilogy(Omf,fftshift(abs(fft(disp))));
title('Predicted Displacement vs.  Actual');
legend('Actual','Kalman','Static');

figure(3)
clf
loglog(Omf,fftshift(abs(fft(errst))),'g')
hold on
semilogy(Omf,fftshift(abs(fft(disp'))),'b')
semilogy(Omf,fftshift(abs(fft(errkalm))),'r')
std(errkalm)/std(disp')
title('FFT of Errors vs.  FFT of Actual Displacement');
legend('Static','Actual','Kalman');

figure(4)
clf
plot(Omf,fftshift(abs(fft(disp))),'b');
hold on
plot(Omf,fftshift(abs(fft(dispX))),'r');
plot(Omf,fftshift(real(fft(errkalm))),'g')
legend('FFT Displacement','FFT Kalman Est.','FFT Kalman Error');

figure(5)
clf
plot(Omf,fftshift(abs(fft(errkalm))),'r')
```

```
  hold on
  plot(Omf,fftshift(abs(fft(errst))),'g')
end
```

C.4 Kalman Filter Code for Experimental Analysis

```
Zc(1,:)=temp(:,3)'-mean(temp(:,3));  %extracts the strain gage readings
Zc(2,:)=temp(:,4)'-mean(temp(:,4));
Zc(3,:)=temp(:,5)'-mean(temp(:,5));
Zc(4,:)=temp(:,6)'-mean(temp(:,6));
Zc(5,:)=temp(:,7)'-mean(temp(:,7));

comega=omega(1)/3*2*time/length(Zc);
cheat=fir1(1024,comega,'high');
Z=filter2(cheat,Zc);

for jj=1:length(Z);
  Zb(:,jj)=Z(:,length(Z)-jj+1);
end

disp=filter2(cheat,temp(:,8)'-mean(temp(:,8)));
t=temp(:,1);
deltat=t(2)-t(1);
A=AE;
B=BE;
C=CE;
D=0;
used=9;
modes=size(A)/2;
%BB=[B(1:used); B(modes+1:modes+used)];
```

```
%Q=BB*BB'/10;

%Establishes the model

Am=[A(1:used,1:used) A(1:used,modes+1:modes+used)
  A(modes+1:modes+used,1:used) A(modes+1:modes+used,modes+1:modes+used)];
BB=[B(1:used); B(modes+1:modes+used)];
QQ=BB*BB';
AA=[-Am QQ; zeros(size(Am)) Am']*deltat;
BC=expm(AA);
Phi=expm(Am*deltat);
Q=Phi*BC(1:used*2,used*2+1:used*4);
Q=(Q+Q')/2;

%set initial conditions
Po=[.0345 .1116 .0328 .0075 .0075 .0075 .0075 .0075 .0075]; eye(used*2);
Po=Po(1:used);
%Po=diag(Q(10:18,10:18))'./(omega(1:9).^2);
P=diag([Po Po.*(omega(1:used).^2)]);

Xhat=zeros(used,length(t));
Xhatdot=zeros(used,length(t));
Am=[A(1:used,1:used) A(1:used,modes+1:modes+used)
    A(modes+1:modes+used,1:used) A(modes+1:modes+used,modes+1:modes+used)];
H=[CE(2:6,1:used) zeros(5,used)];
h=H(1:5,1:5);
omegab=omega(L)*time/length(Z)*2;
N=2^ceil(log2(1/(omega(1)*time/length(Z)*2)));
if omegab>1
  omegab=.999;
end
ZF=fir1(N,omegab);
Z2=Z-filter2(ZF,Z);
R=diag([1 1 1 1 1])*mean(std(Z2'))*10;

diag(std(Z')*.05);diag([10 5 5 5 5])*.1;
```

```
Phi=expm(Am*deltat);

Out=0;
Outz=0;

%Runs the filter foward once
for T=2:length(t)
   Pm=Phi*P*Phi'+Q;
   P=(eye(used*2)-Pm*H'*inv(H*Pm*H'+R)*H)*Pm;
   K=Pm*H'*inv(H*Pm*H'+R);                                          70
   TT=Phi*[Xhat(:,T-1); Xhatdot(:,T-1)];
   Xhatm=TT(1:used);
   Xhatdotm=TT(used+1:used*2);
   TT2=TT+K*(Z(:,T)-H*TT);
   Xhat(:,T)=TT2(1:used);
   Xhatdot(:,T)=TT2(used+1:used*2);
end
Xhat1=Xhat;
%1
Xhatb(:,1)=Xhat(:,length(Z));                                       80
Xhatdotb(:,1)=-Xhatdot(:,length(Z));

%Reverses the filter to try to obtain better intial values

for T=2:length(t)
   TT=Phi*[Xhatb(:,T-1);Xhatdotb(:,T-1)];
   Xhatm=TT(1:used);
   Xhatdotm=TT(used+1:used*2);
   TT2=TT+K*(Zb(:,T)-H*TT);
   Xhatb(:,T)=TT2(1:used);                                          90
   Xhatdotb(:,T)=TT2(used+1:used*2);
end
for jj=1:length(Z);
  Xhat2(:,jj)=Xhatb(:,length(Z)-jj+1);
  Xhatdot2(:,jj)=Xhatdotb(:,length(Z)-jj+1);
end
```

```
Xhat(:,1)=Xhatb(:,length(Z));
Xhatdot(:,1)=-Xhatdotb(:,length(Z));

%Runs the filter foward again
%2
for T=2:length(t)
   TT=Phi*[Xhat(:,T-1);Xhatdot(:,T-1)];
   Xhatm=TT(1:used);
   Xhatdotm=TT(used+1:used*2);
   TT2=TT+K*(Z(:,T)-H*TT);
   Xhat(:,T)=TT2(1:used);
   Xhatdot(:,T)=TT2(used+1:used*2);
end
```

Appendix D

Matlab Experiment Code

D.1 Experiment Code

The following is the code that was used to perform the Experimental Analysis, including the Quasi-Static Estimation and the Filtered Cases.

```
load tf^M
load longcap^M
data=data(1:8192*5,:);^M
omega=omega/2/pi;^M
L=length(omega)-1;^M
LL=length(data);^M
%bh=1/temp(LL,1);^M
used=8;^M
t=(1:length(data))/2560;^M
time=max(t);^M                                                    10
Om(1)=0;   %Om is the frequencies for the fft^M
 at10.0ptfor jj=1:LL/2^M
  Om(jj+1)=(jj)/time;   ^M
  Om(LL+1-jj)=-Om(jj+1);^M
end^M
Omf=fftshift(Om);  % use this with the fftshift of the fft for more clarity^M
tdata=fft(data);^M
%maketf;^M
```

```
filts=ones(length(data),1);%ones(size(g))./fftshift(abs(g));ones(length(data),1);^M
[duh,I]=min((abs(Om-omega(1)/3)));^M
patc=linspace(0,1,I);^M
filts(1:I)=patc;^M
filts(length(data)-I+1:length(data))=1-patc;^M
filts(:,2)=filts;^M
filts(:,3:4)=filts;^M
filts(:,5:8)=filts;^M
^M
temp=real(ifft(tdata.*filts));^M
clear data tdata filts^M
^M
expkalmimproved;  %Kalman Filter M-File^M
^M
H=[CE(2:6,1:used) zeros(5,used)];^M
hh=inv(H(1:5,1:5));^M
Xs=zeros(5,LL);^M
Zt=Z';^M
clear QB^M
Xs=hh*Z;  %Creates Static Estimate^M
^M
^M
% This loop creates the filters with corner frequencies at the^M
% Geometric mean of the adjacent modes^M
^M
for jj=1:5^M
  W(jj)=[sqrt(omega(jj)*omega(jj+1))]*time/length(Z)*2;^M
  [QB(jj,:)]=fir1(25,W(jj));^M
end^M
^M
z=fft(Z')';^M
W2=[sqrt(omega(5)*omega(6))];^M
^M
Xf=hh*filter2(QB(5,:),Z);^M
^M
for jj=1:5^M
```

```
  Xf2(jj,:)=filter2(QB(jj,:),hh(jj,:)*Z);^M
end^M
^M
stXs=std(disp-sum(Xs));^M
maxXs=max(abs(disp-sum(Xs)));^M
errst=disp'-sum(Xs)'-mean(disp'-sum(Xs)');^M                                    60
^M
dispK=sum(Xhat);%sum(Xhat(1:5,:));^M
dispXs=sum(Xs);^M
dispXf=sum(Xf);^M
dispXf2=sum(Xf2);^M
ref=std(disp(4000:length(disp)-4000));^M
^M
% The following are the output errors for each Method.^M
errK=std(dispK(4000:length(disp)-4000)-disp(4000:length(disp)-4000))/ref^M
errXs=std(dispXs(4000:length(disp)-4000)-disp(4000:length(disp)-4000))/ref^M    70
errXf=std(dispXf(4000:length(disp)-4000)-disp(4000:length(disp)-4000))/ref^M
errXf2=std(dispXf2(4000:length(disp)-4000)-disp(4000:length(disp)-4000))/ref^M
^M
```

D.2 Kalman Code

This code was used to create the Kalman Filter Estimate.

```
^M
%Zc(1,:)=real(ifft(tempf(:,4)./tempf(:,1)))';%-mean(temp(:,4));^M
%Zc(2,:)=real(ifft(tempf(:,5)./tempf(:,1)))';%-mean(temp(:,5));^M
%Zc(3,:)=real(ifft(tempf(:,6)./tempf(:,1)))';%-mean(temp(:,6));^M
%Zc(4,:)=real(ifft(tempf(:,7)./tempf(:,1)))';%-mean(temp(:,7));^M
%Zc(5,:)=real(ifft(tempf(:,8)./tempf(:,1)))';%-mean(temp(:,8));^M
^M
Zc(2,:)=temp(:,5)'-mean(temp(:,5));^M
Zc(3,:)=temp(:,6)'-mean(temp(:,6));^M
Zc(1,:)=temp(:,4)'-mean(temp(:,4));^M
Zc(4,:)=temp(:,7)'-mean(temp(:,7));^M
Zc(5,:)=temp(:,8)'-mean(temp(:,8));^M
^M
Z=Zc;^M
clear Zc^M
^M
mp(:,3)));^M
disp=temp(:,3)'-mean(temp(:,3));^M
^M
5^M
deltat=t(2)-t(1);^M
A=AE;^M
B=BE;^M
C=CE;^M
D=0;^M
used=9;^M
modes=size(A)/2;^M
^M
Am=[A(1:used,1:used) A(1:used,modes+1:modes+used)^M
  A(modes+1:modes+used,1:used) A(modes+1:modes+used,modes+1:modes+used)];^M
BB=[B(1:used); B(modes+1:modes+used)];^M
```

```
QQ=BB*BB';^M
AA=[-Am QQ; zeros(size(Am)) Am']*deltat;^M
BC=expm(AA);^M
Phi=expm(Am*deltat);^M
Q=Phi*BC(1:used*2,used*2+1:used*4);^M
Q=(Q+Q')/2*.1;^M
^M
Po=[.0345 .1116 .0328 .0075 .0075 .0075 .0075 .0075 .0075]; eye(used*2);^M
Po=Po(1:used);^M                                                          40
P=diag([Po Po.*(omega(1:used).^2)]);^M
^M
Xhat=zeros(used,length(t));^M
Xhatdot=zeros(used,length(t));^M
Am=[A(1:used,1:used) A(1:used,modes+1:modes+used)^M
   A(modes+1:modes+used,1:used) A(modes+1:modes+used,modes+1:modes+used)];^M
H=[CE(2:6,1:used) zeros(5,used)];^M
h=H(1:5,1:5);^M
omegab=omega(L)*time/length(Z)*2;^M
N=2^ceil(log2(1/(omega(1)*time/length(Z)*2)));^M                          50
if omegab>1^M
  omegab=.999;^M
end^M
ZF=fir1(N,omegab);^M
Z2=Z-filter2(ZF,Z);^M
R=diag([1 1 1 1 1])*mean(std(Z2'))*15;^M
^M
%diag(std(Z')*.05);diag([10 5 5 5 5])*.1;^M
Phi=expm(Am*deltat);^M
^M                                                                        60
Out=0;^M
Outz=0;^M
^M
for T=2:length(t)^M
   Pm=Phi*P*Phi'+Q;^M
   P=(eye(used*2)-Pm*H'*inv(H*Pm*H'+R)*H)*Pm;^M
   K=Pm*H'*inv(H*Pm*H'+R);^M
```

```
    TT=Phi*[Xhat(:,T-1); Xhatdot(:,T-1)];^M
    Xhatm=TT(1:used);^M
    Xhatdotm=TT(used+1:used*2);^M                                    70
    TT2=TT+K*(Z(:,T)-H*TT);^M
    Xhat(:,T)=TT2(1:used);^M
    Xhatdot(:,T)=TT2(used+1:used*2);^M
    if mod(T*20,length(t))==0^M
      round(T/length(t)*100)^M
    end^M
end^M
```

Bibliography

[1] Mark S. Andersson and Edward F. Crawley. Discrete distributed strain sensing of intelligent structures. In *Second Joint Japan/US Conference on Adaptive Structures*, Nov 1991.

[2] Mark S. Andersson and Edward F Crawley. Structural shape estimation using shaped sensors. In *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, apr 1995.

[3] Mark Sven-Erik Andersson. Spatially filtering strain sensors for structural shape estimation of intelligent structures. Master's thesis, Massachusetts Institute of Technology, February 1993.

[4] A. Baz and S. Poh. A new class of distributed sensors. In *Transactions of the ASME*, 1997.

[5] Robert Grover Brown and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering.* John Wiley and Sons, 1997.

[6] Robert Grover Brown and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering*, chapter 5.3. John Wiley and Sons, 1997.

[7] Shawn E. Burke and James E. Hubbard Jr. Distributed actuator control design for flexible beams. *The Journal of IFAC the International Federation of Automatic Control*, pages 619–627, Sept 1988.

[8] F. Charette and A. Berry. Active control of sound radiation from a plate using a polyvinylidene fluoride volume displacement sensor. *Journal of the Acoustical Society of America*, 1998.

[9] W. K. Chiu, M. Heller, and R. Jones. Determination of the stress components of an array of piezoelectric sensors: A numerical study. *Smart Materials and Structures*, 1997.

[10] Li Debao, Zhuge Hongcheng, and Wang Bo. The principle and techniques of experimental strain modal analysis. In *Proceedings of the 7th International Modal Analysis Conference*, 1989.

[11] Stephen J. Fonash. Integration of thin film electronics and MEMs. In *SPIE Vol. 2722*, 1996.

[12] G.C. Foss and E.D. Haugse. Using modal test results to develop strain to displacement transformations. Technical report, Boeing, 1992.

[13] Michael Fripp. 2000. Will appear in ICAST Proceedings for 2000.

[14] Nesbitt W. Hagood, Walter Chung, and Andreas von Flotow. Modeling of piezoelectric actuator dynamics for active structural control. *Journal of the Acoustical Society of America*, 1998.

[15] Nesbitt W. Hagood and Andreas von Flotow. Damping of structural vibrations with piezoelectric material and passive electrical networks. *Journal of Sound and Vibration*, 1991.

[16] J. H. Han and I. Lee. Active damping enhancement of composite plates with electrode designed piezoelectric materials. *Journal of Intelligent Material Systems and Structures*, 8(3):249–259, 1997.

[17] Charles Hautamaki, Shayne Zurn, Susan Mantell, and Dennis Polla. Embedded microelectromechanical systems (MEMS). Technical report, University of Minnesota, 1998.

[18] James K Henry and Robert L. Clark. A model of a curved piezo-structure for active structural acoustic control. In *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit*, number 39, pages 3223–3231, Long Beach CA, apr 1998. AIAA, Academic Press.

[19] S. Y. Hong, V. V. Varadan, and V. K. Varadan. Prediction of modal responses for optimal selection of actuator positions in rectangular plate. In *SPIE Vol. 2189*, 1994.

[20] M. Hopkins and J. Tuss. Smart skin conformal load-bearing antenna and other smart structures development. In *AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and Exhibit*, apr 1997.

[21] George C. Kirby, Tae W. Lim, R. Webere, A. B. Bosse, C. Povich, and S. Fisher. Strain-based shape estimation algorithms for a cantilevered beam. In *SPIE Vol. 3041*, 1997.

[22] Jayanth N. Kudva, Allen J. Lockyer, and Craig B. Van Way. Structural health monitoring of aircraft components. In *Smart Materials and Structures: Implications for Military Aircraft of New Generation*, 1996.

[23] C. K. Lee and F. C. Moon. Modal sensors/ actuators. In *Transactions of the ASME*, 1990.

[24] Mark Lin and Fu-Kuo Chang. Development of smart layer for built-in diagnostics for composite structures. In *Proceedings of the 13th Annual Technical Conference of Composite Materials*, Baltimore, Maryland, Sept 1998.

[25] A.J. Lockyer, J.N. Kudva, B.P. Hill D.M. Kane, and A. Martin. A qualitative assessment of smart skins and avionic/structures integration. In *SPIE Vol. 2189*, 1994.

[26] D. W. Miller, S. A. Collins, and S. P. Peltzman. Development of spatially convolving sensors for structural control applications. In *Proceedings of the 31st*

AIAA/ASME/ASCE/AHS Structures, Structural Dynamics and Materials Conference, Apr 1990.

[27] D. J. Pines and Andreas H. von Flotow. Active control of bending wave propagation at acoustic frequencies. *Journal of Sound and Vibration*, 1990.

[28] A. Priou. Exploratory team on smart structures. In *Smart Materials and Structures: Implications for Military Aircraft of New Generation*, 1996.

[29] John P. Rodgers. *Development of an integral twist-actuated rotor blade for individual blade control.* PhD thesis, Massachusettes Institute of Technology, 1997.

[30] Bob S. Shen, Mike Tracy, Youn-Seo Roh, and Fu-Kuo Chang. Built-in piezoelectrics for processing and health monitoring of composite structures. In *AIAA/ASME/AHS Adaptive Structures Forum*, Salt Lake City, UT, apr 1996.

[31] W. Shields, J. Ro, and A. Baz. Control of sound radiation from a plate into an acoustic cavity using active piezoelectric-damping composites. *Smart Materials and Structures*, 1998.

[32] L. M. Vári and P. S. Heyns. Using strain modal testing. In *Proceedings of the 12th International Modal Analysis Conference*, 1994.

[33] David J. Warkentin. Embedded electronics for intelligent structures. Master's thesis, Massachusetts Institute of Technology, 1991.

[34] David J. Warkentin. *Power amplification for piezoelectric actuators in controlled structures.* PhD thesis, Massachusetts Institute of Technology, 1995.

[35] David J. Warkentin, Edward F. Crawley, and Stephen D Senturia. Embedded electronics for intelligent structures. Technical report, Massachusetts Institute of Technology, 1991.