MEASUREMENTS OF WEB DEFLECTION FOR THE DETERMINATION OF STABILITY OF A WEB

By

XIANGMING FANG

Bachelor of Science

East China University of Chemical Technology

1991

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 1999

OKLAHOMA STATE UNIVERSITY

MEASUREMENTS OF WEB DEFLECTION FOR THE

DETERMINATION OF STABILITY OF A WEB

Thesis Approved:

Peter M Moretti

Thesis Adviser

Young Bae Chang

Hongbing Lu

Wayne B Powell

Dean of the Graduate College

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my major adviser, Dr. Peter Moretti for his intelligent supervision, constructive guidance, and invaluable support during my research and study. Also, I would like to thank Dr. Hongbin Lu and Dr. Y.B. Chang for their kind help from the beginning of my coming to OSU to the completion of my research.

I would like to give my special appreciation to my wife, who helped me to write computer code.

Finally, I would like to thank the Web Handling Research Center and the School of Mechanical and Aerospace Engineering for providing me with this research opportunity and financial support.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

NOMENCLATURE

λ	Wave length of web path
ξ	x – direction coordinate moving with belt
ω_n	Angular frequency
ρ	Density of air
μ	Density of fluid
c	Wave speed
D	Bending Stiffness, D = EI
EI	Bending stiffness of air-supported web
g_c	Gravity unit conversion factor
h	The flotation height of the web over an airbar
$Intensity_1, \ldots, Intensity_n$	The intensity value of $Z_1, Z_2, \ldots Zn$
M_{web}	Mass of web
M_a	Hydrodynamic mass coefficient of $\partial^2 z/\partial t^2$
M_b	Hydrodynamic mass coefficient of $\partial^2 z/\partial x \partial t$
M_c	Hydrodynamic mass coefficient of $\partial^2 z/\partial x^2$
P_{gage}	Gage air pressure
R	Radius of web path
t	time

T	Tension of web
u	Running speed of web
z	Vertical displacement of web
Z_{CENTER}	The position of the center of the web line
$Z_1, Z_2, \ldots Zn$	The position of the neighbor of the center line

CHAPTER 1

INTRODUCTION

Background

A web is a thin continuous strip of material such as paper, plastic film, or fabric that can be wound in a roll. Before it reaches its final stages such as newspaper, plastic wrap, etc., it may undergo processes such as printing, coating and drying. Therefore, strips of substantial length have to be handled. Generally, these webs are moving at high speed to achieve high productivity. In many processes the web is floating on air conveyances.

Air conveyance is the support of a web on non-contacting air cushion devices [20]. There are three kinds of air conveyances:

1) Air bars (low wrap): Air bars include pressure pad air bars (wrap: 2 to 15 degrees) and Bernoulli air bars (wrap: -10 to 0 degrees).
2) Air "reversers" (high wrap): The typical wrap angles of air reversers are from 45 to 180 degrees.
3) Air turning bars (helical wrap): These air bars are typically 180 and 90 degrees wrap web turn.

Air conveyance (see Figure 1-2) is used to direct and dry a web because:

(1) It permits non-contact conveyance of delicate or tacky webs.

(2) It can eliminate mechanical roller problems.

(3) It can eliminate roller-generated waste, scratches, wrinkles etc.

(4) It can eliminate roller-generated electrostatic problems.

(5) It can reduce tension loss and the number of machine driving motors.

(6) It permits additional web line options [20].

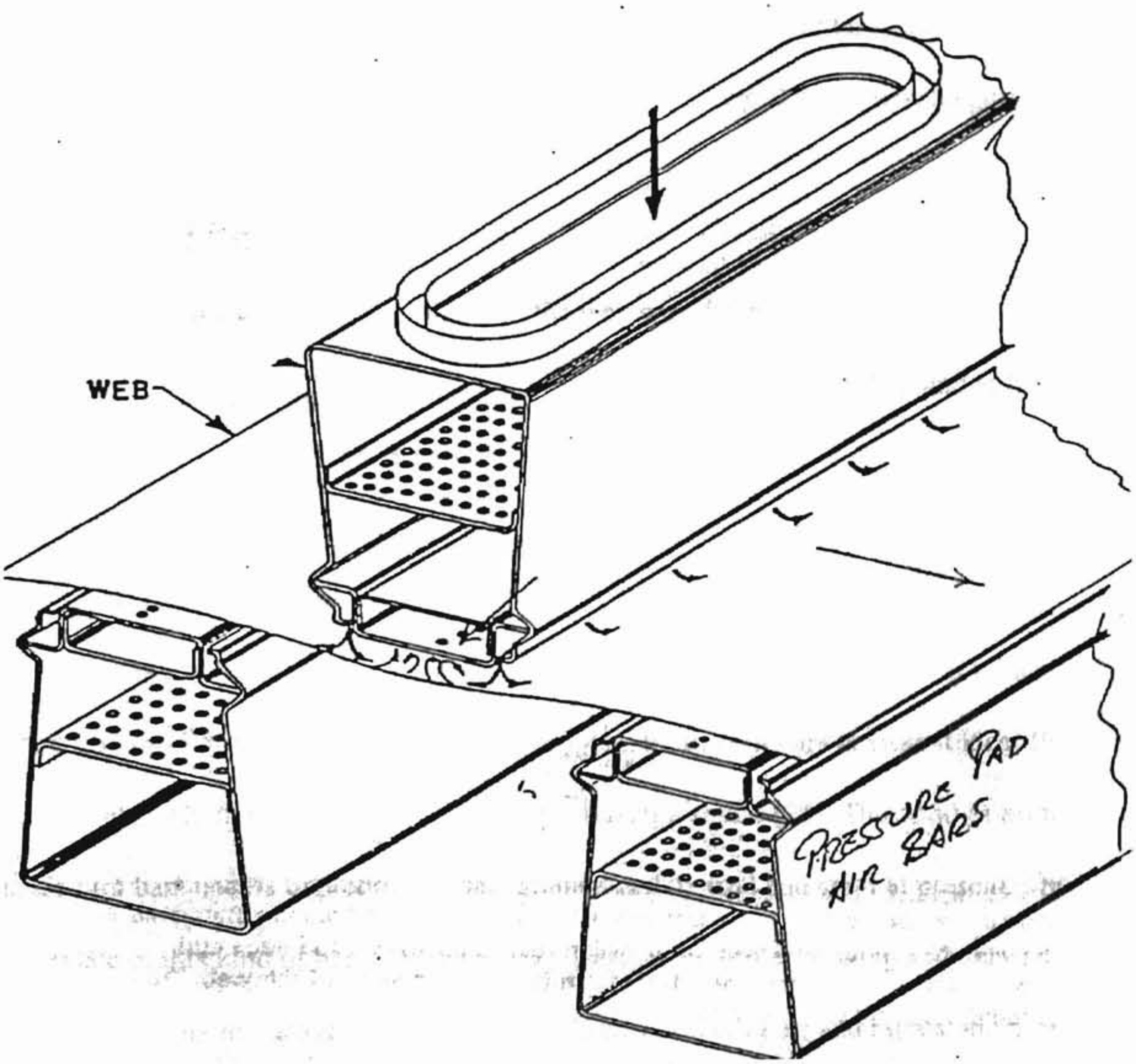

Figure 1-1 Scheme of a Running Web on Air Conveyance

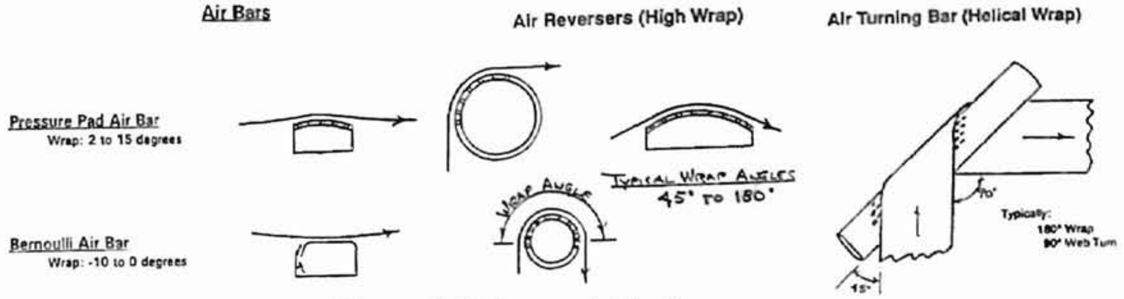

Figure 1-2 Types of Air Conveyance

On the other hand, air conveyance may add to the web flutter problems. The web flutter is influenced by the web speed and airflow. There are at least two categories of flutters in the web handling.

1) Global flutter

In the paper and printing industry, the web speed can reach 6000 fpm. This high speed can cause large aerodynamic force, coupled with inertial, damping and stiffness forces. When air and the web speeds reach a critical value, the web may sail, billow, oscillate or break.

2) Local flutter

Another type of flutter occurs in the area near air conveyance. This is a local phenomenon. For example, one annoying problem in air-turn bars is violent local flutter of the webs, which is often accompanied by a buzzing sound [25]. This kind of problem at air-turn bars occurs unpredictably and is managed by trial and error at present. The clearance of this kind of bar is typically very large at the center of wrap and very small near the entering and leaving nip points. The high velocity of air and high static pressure adjacent to the nips cause this severe clearance non-uniformity. When the hole pattern extends only to the nip, the web will vibrate/flutter severely in the adjacent span. This phenomenon is caused by the high velocity air exhausting at the nip. The Bernoulli pressure reduction pulls the web toward the reverser beyond the hole pattern, resulting in a reed-like vibration.

Problem Statement

As mentioned above, there are basically two different types of flutter problems in the web industry. The most important factor in these problems is the aerodynamic force acting on the web. It is very useful to study the flutter by detecting the position and shape of the web.

The analytical model for steady motion of a web can be simplified as follows:

$$\left(\frac{\mu}{g_c}u^2 - T\right)\frac{\partial^2 z}{\partial x^2} + EI\frac{\partial^4 z}{\partial x^4} = P_{gage}(x) \qquad \text{(Eq.1-1)}$$

Known: The web velocity, u

The web material and property, μ, E, I, etc.

The web tension, T

The web flotation height, z

Determine: Pressure acting on the web, P_{gage}.

In order to know the pressure acting on the web caused by air bar, we need to know the web shape. Then, from the web shape we can find the web curvature $\partial^2 z / \partial x^2$ and the fourth derivative $\partial^4 z / \partial x^4$. If the web is flexible and moving at a known speed, it is possible to calculate the pressure distribution from the shape.

Therefore, the problem here is how to obtain the web shape. In this project, the objective is to get the web shape from a digital camera image.

CHAPTER 2

LITERATURE REVIEW

Analytical Solution of the Web Deflection

Since the web deflection is so important for us to get pressure profile, we need to study the analytical solution for air-web interaction.

Lighthill (1960) derived an equation for swimming fish. This is basically the original model of the "traveling thread line model" used in the web handling. In this model, Lighthill assumed stationary fluid. For the observer moving with the web, the governing equation for a web surrounded by air becomes [10]:

$$(M_{web} + M_a)\frac{\partial^2 z}{\partial t^2} \pm 2M_b u\frac{\partial^2 z}{\partial \xi \partial t} + (M_c u^2 - T)\frac{\partial^2 z}{\partial \xi^2} + EI\frac{\partial^4 z}{\partial \xi^4} = f(\xi, t) \qquad \text{(Eq.2-1)}$$

In the above equation, first term is the system acceleration. Because this system includes the air moving with the web, there is a mass of air (M_a) involved. At low speed, this added mass could be found by integrating the kinetic energy caused by a reference out-of-plane velocity of the web in the two dimensional flow field, and then setting the results equal to one-half times M_a times the square of the reference velocity [10].

The second term is a Coriolis term, positive if the ξ coordinate is taken opposite to the motion of the M_b. T is the web tension of the web. The fourth term is the centrifugal force.

The right hand side term is the pressure difference across the web.

By using the above model and neglecting bending stiffness EI, the n^{th} mode complementary solution for a span of length L is expressed as [10]:

$$z = \cos(\omega_n t + \frac{n\pi x}{L}\frac{u}{c})\sin(\frac{n\pi x}{L}) \qquad \text{(Eq.2-2)}$$

Where: c is the wave speed, $c = \sqrt{\dfrac{T}{M_{web} + M_a}}$

$$\omega_n = \frac{n\pi c}{L}[1-(u/c)^2]$$

u is the web moving speed in machine direction.

Furthermore, they studied the aerodynamic terms of the web and found the divergent and flutter speed.

Dowell [14] studied a thin plate elastic panel supported at its edge. He considered the simple case and studied the "panel flutter" effect. Bolotin built a relatively complex model, provided by a plate lying in an elastic foundation and exposed to a flow of gas. He considered the aerodynamic non-linearity effects and found that the solution depends on stiffness parameter [5].

Muftu developed Bolotin's model to study another flutter that occurred in helical-scan recording. He used the model of moving cylinder shell to build the governing equation of a magnetic film [18, 26]. The equation is:

$$\begin{aligned} &D\nabla^4 w + (\rho c V_x^{\,2} - T_x)\frac{\partial^2 w}{\partial x^2} + kw + 2\rho c V_x \frac{\partial^2 w}{\partial x \partial t} + \rho c \frac{\partial^2 w}{\partial t^2} \\ &= P_a + P_0 e^{-\alpha[x - x_0(t)]^2} e^{-\alpha[y - y_0(t)]^2} \end{aligned} \qquad \text{(Eq.2-3)}$$

D is the bending stiffness, ∇^4 is the biharmonic operator, w is the radius displacement. x,y are circumferential and axial coordinates on the tape. The right-hand side (RHS) is the aerodynamic force and contact pressure.

The above model is three-dimensional and is solved numerically. Without aerodynamic force, the critical load speed of a point load is identified by investigating the flexural waves in the tape. Muftu discovered that the critical load speed is proportional to tape tension, tape thickness and tape transport speed. His study also showed that there is more "free edge flutter" in thicker tapes than in thinner ones.

Generally, the P_a of RHS of (Eq.2-3) is very complicated. There is no specific math expression when Mach number $M_\infty \leq 1$. The RHS term depends on the web curvature, moving speed, and moving angle (angle of attack), etc. Normally the RHS terms are coupled with inertia, damping and spring term in every time step.

It is difficult to get the web deflection from pure analysis. In some cases, we can obtained some numerical solutions from commercial CFD code. Direct measurement is another way to solve this problem.

Measurement

Not much literature was found relating to web measurements.

Paper technologist Anson [2] (1995) used fast Fourier transform (FFT) to detect the presence of periodic marks in paper. Dewitte [13] used a video surveillance system to pinpoint where breaks occurred. His system consisted of a synchronizing system, a video imaging system and a computer-controlled recording and playback system. This system has been beneficial in detecting the web locations at a paper mill.

Budd [6] (1996) introduced binary morphology for recognition of objects by shape, applied grayscale morphology to improve signal and noise ratio, and used array sensors and parallel processors for high-speed inspections. He found when the differential between the defect gray level and the background gray level is large (>20 grayscale values), the global threshold will generate acceptable results; but uniform illumination of a curved surface is very difficult to get.

Nordstrom, et al. [27] (1997), obtained the first curvature measurement found in the literature. They measured the curvature of thin paper. Their method is based on measurement of the out-of-plane deflection of a curled sheet with non-contact laser distance meter. It is interesting to see that the accuracy of the test procedure is mostly governed by the accuracy of the deflection measurement, which in turn is governed by the resolution of the optical distance meter (50μm). Another result was that the relation $\partial^2 z / \partial x^2 = 1/R$ is valid only locally when the overall deflections are large.

Another measurement using optical method is developed by Berndston and Niemi [4]. They used an "optoelectronic camera"(digital camera) to record images of the dry line region on the wire of a paper machine under specified illumination. After an image was obtained, it was transmitted to a computer for extraction and display of dry-line data. Their method has been implemented on industrial paper machines in two versions. The dry-line data obtained are correlated with properties of the final product and can therefore be used for automatic control of the paper machine.

In image processing, the measurement of the web line can be done by edge detection. Edge detection is a special digital image filter. Different software gives it different name. For example, Adobe Photoshop® calls it "lasso", PaintShop® calls it edge detection. Commercial software can detect the web line but does not give its math expression.

Specifically first step of the measurement of the web line is tracing the line in an image. Then the suitable curve fitting method should be used to smoothly describe the curve.

First Step: Detecting the Line

In image processing there are several different methods for detecting the line.

One is using the Hough transformation. Hendry, et al. [17] (1988), use Hough filter to locate thin line with arbitrary and variable curvature in synthetic aperture radar (SAR) image. Princen [29] (1990) optimized the Hough transformation filter and gave the relative algorithm. Palmer [28] (1992) wrote a Hough transform algorithm with a 2-D hypothesis-testing kernel. Lynn Abbott [1] (1994) used the SPLASH2 computer system and a Hough transform to find lines. Dehili [12] (1994) presented four approaches to paralleling the Hough Transform on a hierarchical structure called hyper-pyramid. Sugawara [30] improved that transform and used a weighted Hough Transform on a grid image plan. He used it to detect the skew of the document images. Yang, et al. [33] gave another improvement based on a likelihood principle of connectivity and thickness for line detection. His method had better power of line detection in a noisy image.

Other ways were focused on different direction. Verbeek [31] presented a paralleled (filter) method which allows filtering in different directions simultaneously. Merlet and Zerubia [24] used a mathematical formalization of F* algorithm to detect roads and valleys on satellite images. All the needed information (contour, gray-level and curvature) was synthesized in a unique cost function defined on the digital original image. Chen and Yip [9] developed a simple and efficient approach in the line segment detection algorithm, which employed the properties of digital line segment. In discrete domain, a continuous line can be considered as a combination of sets pattern in the quantified direction of edge pixels (in 0, 45, 90 and 135 degrees) with approximately an equal number of pixels. Based on this characteristic, they derived the convergence of a discrete line. Lim and Alder [23] proposed a non-parameter method for detecting lines and curves. Behar [3] studied the contrast techniques for line detection in a correlated noise environment. Wolfgang [32] presented a line detector based on the scale-space theory for discrete signals introduced by Lindeberg.

Busch [8] measured the web shape by using a camera and digitizing the photograph by framegrabber. She drew a line in the center of the web. Based on the characteristic that the web centerline is continuous, she developed a computer code to detect the web line. In her code, she successfully found the web centerline. Then she used this centerline to find the pressure difference between the upper and lower of the web. But when she used finite difference method to find the first and second derivative, she found that the error was quite large. The reason was that "digitizing the pictures leaves only one significant digit to describe the web shape, which results in a great loss of detail, and in crude errors when using these data for further calculations."

Second Step: Curve Fitting

After getting the image of the web line, the next step is curve fitting.

Lee and Ho [21] employed the curve-fitting method to work out the different parameters to measure the strain. First the digital image was processed by noise filtering, then by edge thinning. Finally, different models were used to fit the parametric equation of an ellipse.

Guo and Xu [16] treated the curve-fitting problem as error propagation problem and employed the general Jacobean linearization to analyze it. They found it is valid for two typical noise models presented.

Grennhill and Davies [15] described an approach to the determination of intensity thresholds for image segmentation. It is based on interpolating between two of more sub-optimal thresholds to obtain a single unbiased value. The particular advantage of the approach is that it avoids the extensive computation involved in rigorous modeling of intensity distributions by curve fitting.

Karen and Cooper [19] discussed the representation of closed implicit polynomials of modest degree for curves in 2-D images and surfaces in 3-D range data. They found that the super quadrics are a small subset of object boundaries that are well fitted by these polynomials.

Liang [22] developed a new transform for curve detection, called the Curve Fitting Hough Transform (CFHT). The CFHT is advantageous over the conventional HT and its variants in its high speed, small storage and high parameter resolution. He achieved this by fitting a segment of the curve detected to a small neighborhood of edge points. If the fitting error is less than a given tolerance, the parameters obtained from curve fitting are used to map an edge element to a single point in the parameter space. His study was implemented for straight line and uniform arc.

CHAPTER 3

METHODS OF APPROACH

Measurements of Web

There are several different ways to measure the web deflection. For example, we can use a line of parallel laser sensors to detect a line of points on the web. Or even we can measure the web deflection by contact sensor. Among these available equipment and methods, digital camera is the cheapest and the most widely available.

The sequence for measurements using a digital camera is: first, use the digital camera to get the web line image; second, use a computer code to eliminate the background and get the pixels of the web line; third, use the special technique called "Center-of-Gravity Method" and curve fitting method to get smoothed deflection curve; fourth, convert the dimensions from pixel counts to actual dimensions in inches. After we obtain the web deflection, we can deduce the aerodynamic pressure acting on the web.

1. Web Image by A Digital Camera

We obtain the digital image by means of a digital camera. Thus, it is not necessary to use a framegrabber board.

In Busch's thesis [8], one of the problems is that the resolution of the final image is low---only 372 × 496 pixels. This low resolution led to big measurement errors when digitizing the image. Digital camera can increase the image resolution, reducing the loss of information when digitizing the image.

Basically there are two different ways to increase resolution. One is to increase the depth resolution. That is to say, we increase the gray levels, for example, from 8 bit ($2^8 = 256$ gray level) to 10 bit ($2^{10} = 1024$ gray level), then each pixel has more information.

Another way is to increase spatial resolution. That is to say, we increase the total number of pixels in an image, for example, from 240 × 320 pixels to 480 × 640 pixels, or even 960 × 1280 pixels.

Most computers available today operate on images that have a depth resolution not greater than 8 bit, but improved spatial resolution is widely available in both digital camera and computer software.

To get a high-resolution image, it is best to let the background have a strong contrast with our object, the web. This makes it easier to detect the web shape. Furthermore, we also keep the camera at long range, or we must consider perspective effects.

2. Computer Code for Detecting the Web Line

There are two possible ways for us to detect the web line and describe it by a mathematical expression.

First we can find commercial software that can detect the web line and describe it by a mathematical expression.

Alternatively we can write a code that can detect the web line and find a good method for fitting the curve. We can use statistical software to analyze the data and fit that curve. Furthermore, only by this can we realize real time web curvature detection.

Software processing will have the following steps:

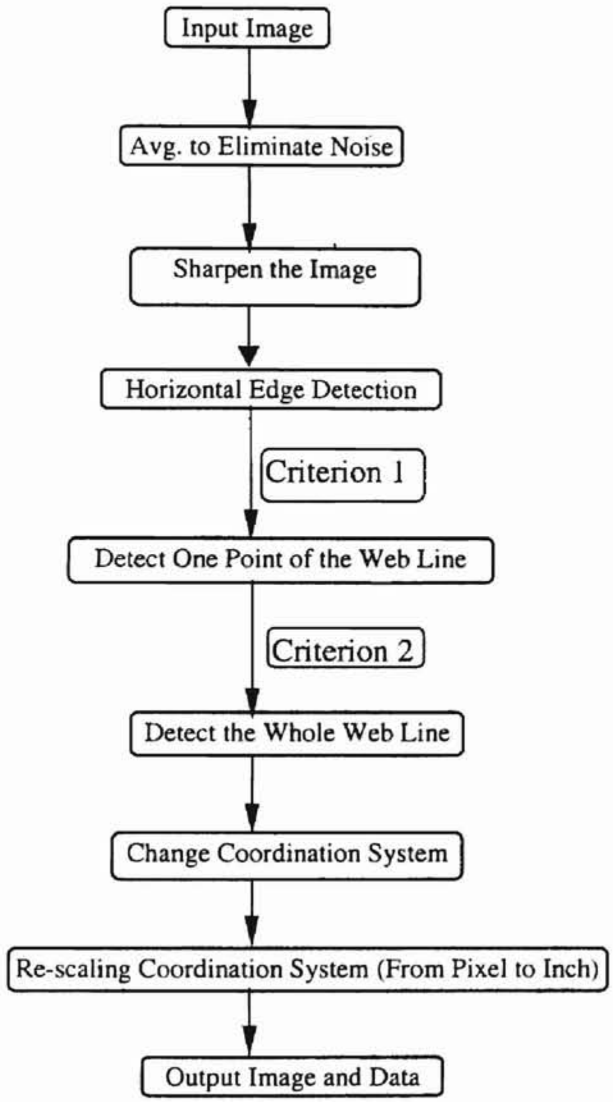

Figure 3-1 Flow Chart of Code to Process the Web Image

The underlying principle of the code to detect the web line is as follows:

(1) Find the center point of the web line.

Since the center point of a web line is located roughly in the middle of the image. it is relatively easy to detect this point. To find this point, we use the property that the white points' intensity is bigger than other points. In this case, the range of search is about 30-100 pixels depending on situation.

(2) Get the rest of the web line.

After locating the first point, it is easier to find the next ones, because the web line is continuous and it is not necessary to use 30-100 pixels of searching range to search again. By using 3-7 pixels range, we can acquire the web line.

After the processing, the web line we get is as follows:

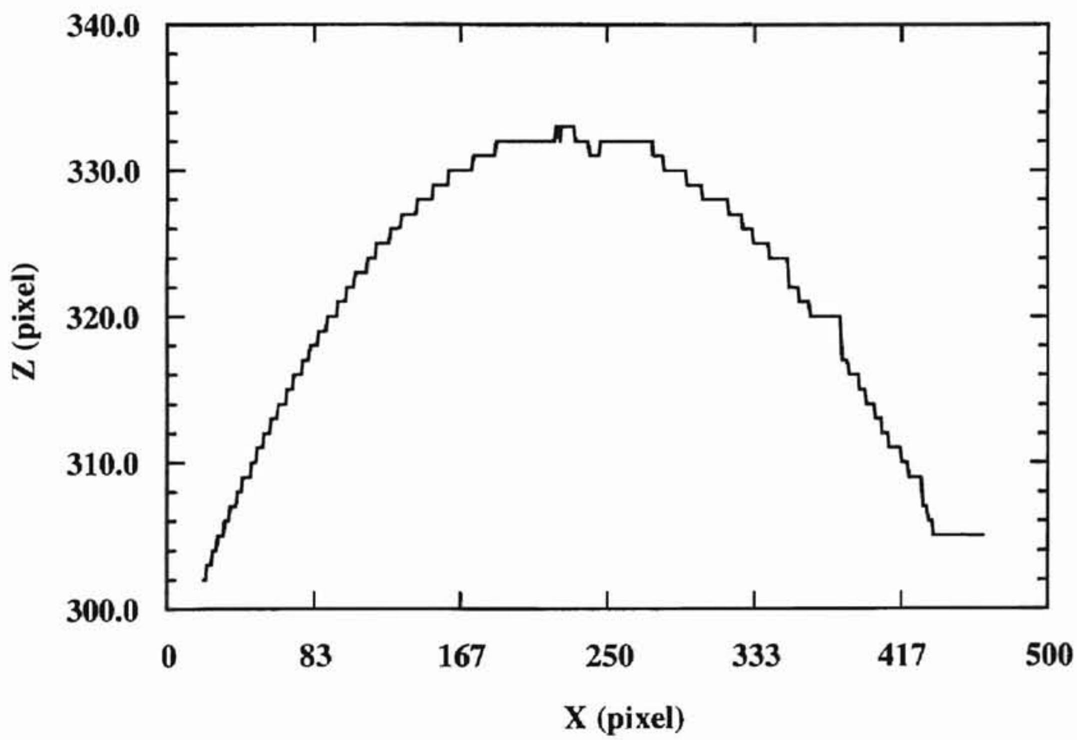

Figure 3-2 The Web Line Detected by Code

From the above figure, we can see that this result is better than the one with the framgrabber. The number of horizontal pixels has increased from 360 to about 620, and the number of vertical pixels from 30 to 40 approximately.

We also see that the vertical resolution of the image is more important than the horizontal. If we change the direction and rotate the image, the result is as follows:

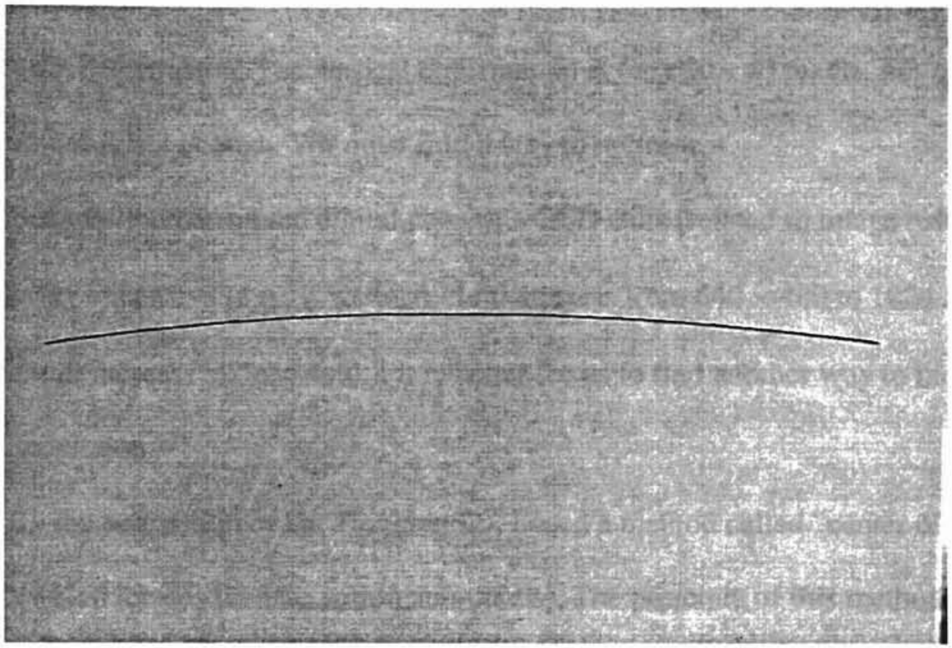

Figure 3-3 Image of the Test Line

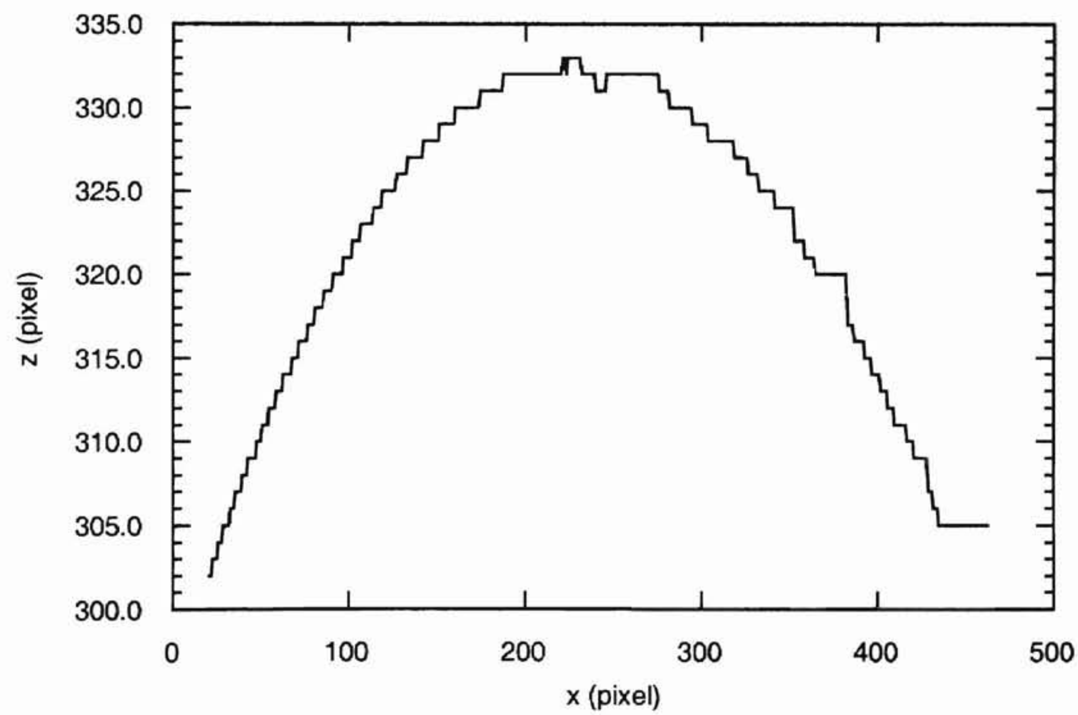

Figure 3-4 The Result Obtained from Rotated Image

3. Center of Gravity Method for Increasing Vertical Resolution

From the above figure, we can see the vertical resolution is extremely important to detect the deformation. The vertical deformation is only 30 - 40 pixels. For our application, it is not enough. We must find a way to increase it.

Because increasing the digital camera's CCD cells (related to image resolution) is prohibitively expensive (e.g., if we increase the pixels from 640×480 to 1024×1024, the price will increase sixteen-fold.), it is better for us to find another way to increase vertical resolution.

By the suggestion of Dr. Hongbin Lu, I used a method called "center of gravity"(which locates the line within sub-pixels). The principle of this method is in Figure 3-5.

In an image file, it is rare that the web line occupies only one pixel. A web line usually occupies 3 – 5 pixels and there is no single sharp edge between the web line and the background. We can see the intensity values around the web line do not increase or decrease abruptly. For example, the intensity value of the image shown in Figure 3-5 is listed in Figure 3-6.

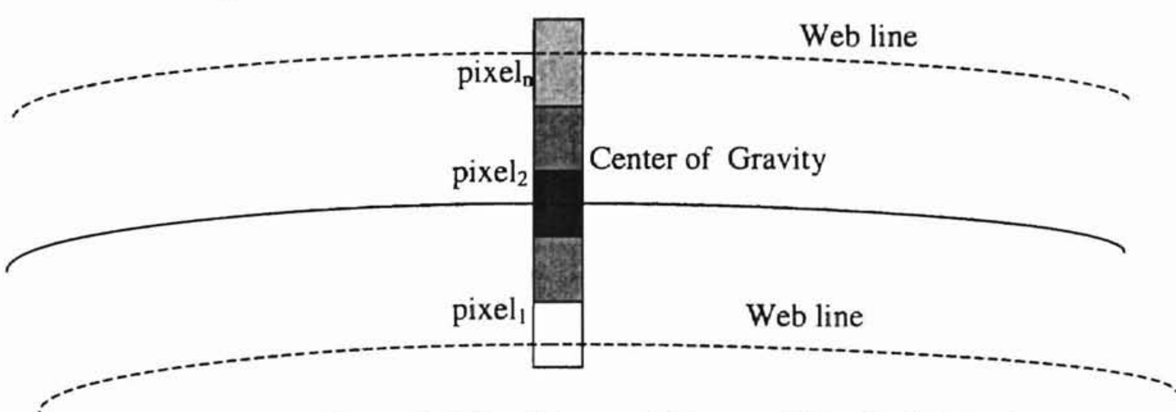

Figure 3-5 The Scheme of Center of Gravity Method

128
146
152
142
134

Figure 3-6 The Value of The Web Line's Intensity

It is difficult for us to decide how many pixels the web line exactly occupies. However, we can get the approximate number by checking the image. After we get the vertical pixel number, it is desired to use the following equation to decide where the "center of gravity" is:

$$Z_{CENTER} = \frac{Z_1 Intensity_1 + Z_2 Intensity_2 + ... + Z_n Intensity_n}{Intensity_1 + Intensity_2 + ...Intensity_n} = \frac{\sum_{i=1}^{n} Z_i Intensity_i}{\sum_{i=1}^{n} Intensity_i} \quad \text{(Eq.3-1)}$$

Where: Z_{CENTER} is the position of the center of the web line.

Z_1, Z_2, …, Z_n stand for the position of the neighbors of the center of that line

$Intensity_1$, $Intensity_2$, … , $Intensity_n$ are the intensity value of Z_1, Z_2, …, Z_n .

4. Curve Fitting

The digital image processed by the computer code is still discrete even though the image resolution is better. However, we can use curve fitting to define a continuous web line. For example, the image Busch [8] got is in Fig 3-7. From Figure 3-7, we can see that the apparent web path has steps. To eliminate them, we use a curve fitting method.

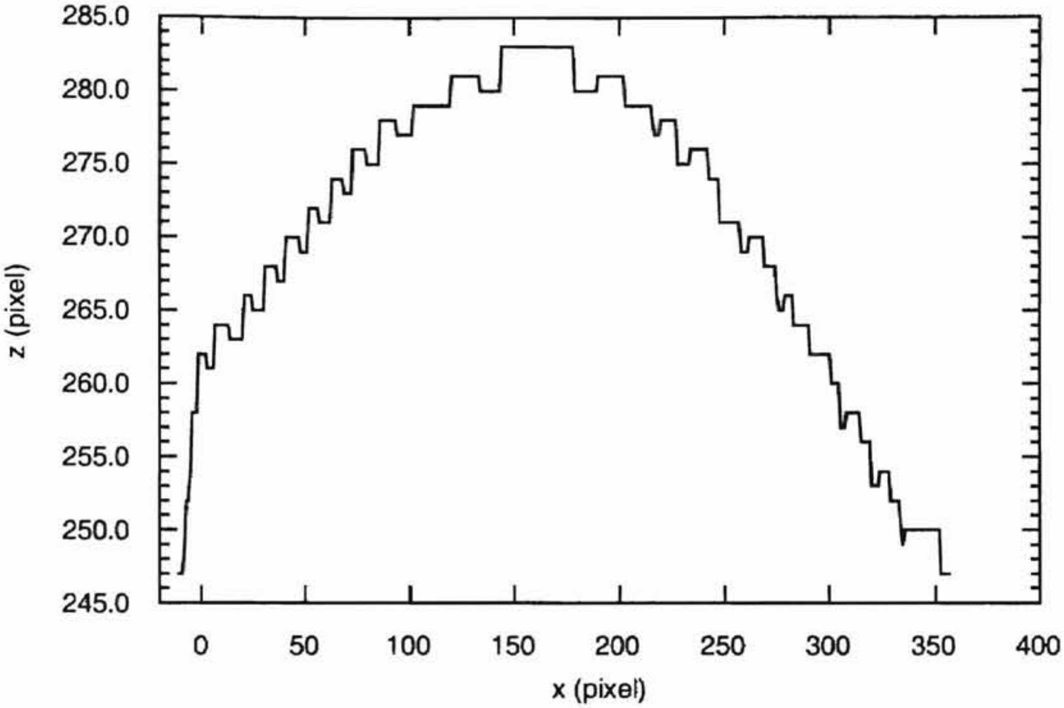

Figure 3-7 The Web Line after Processing

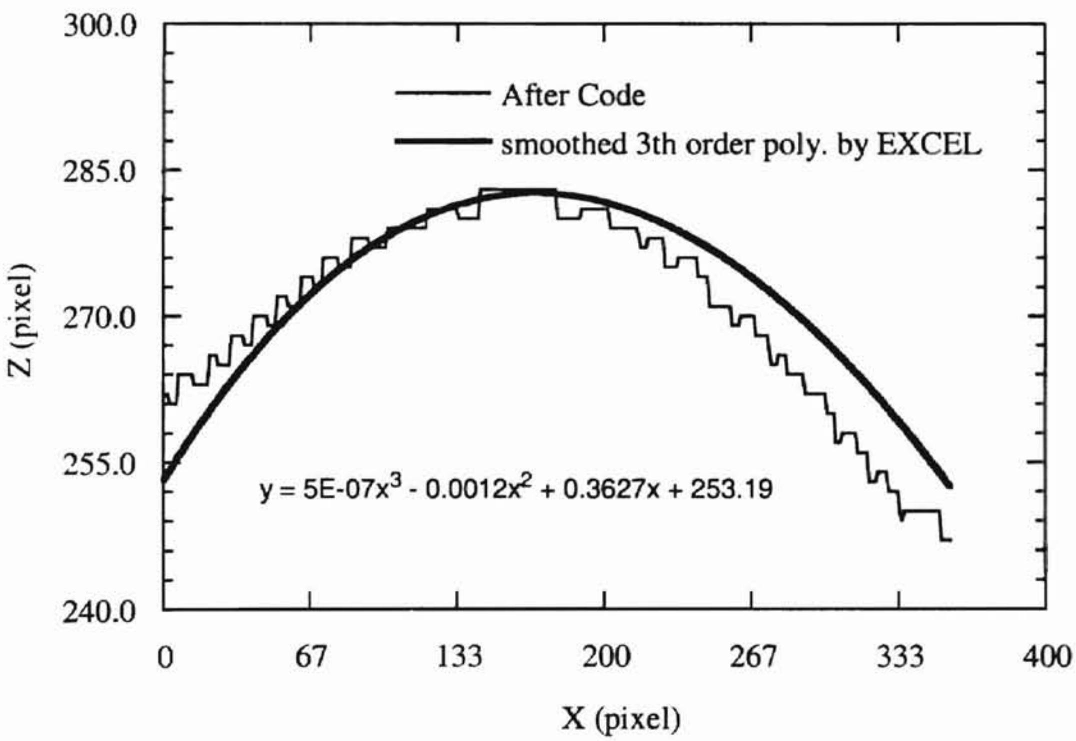

Figure 3-8 3rd Order Polynomial Curve Fitting

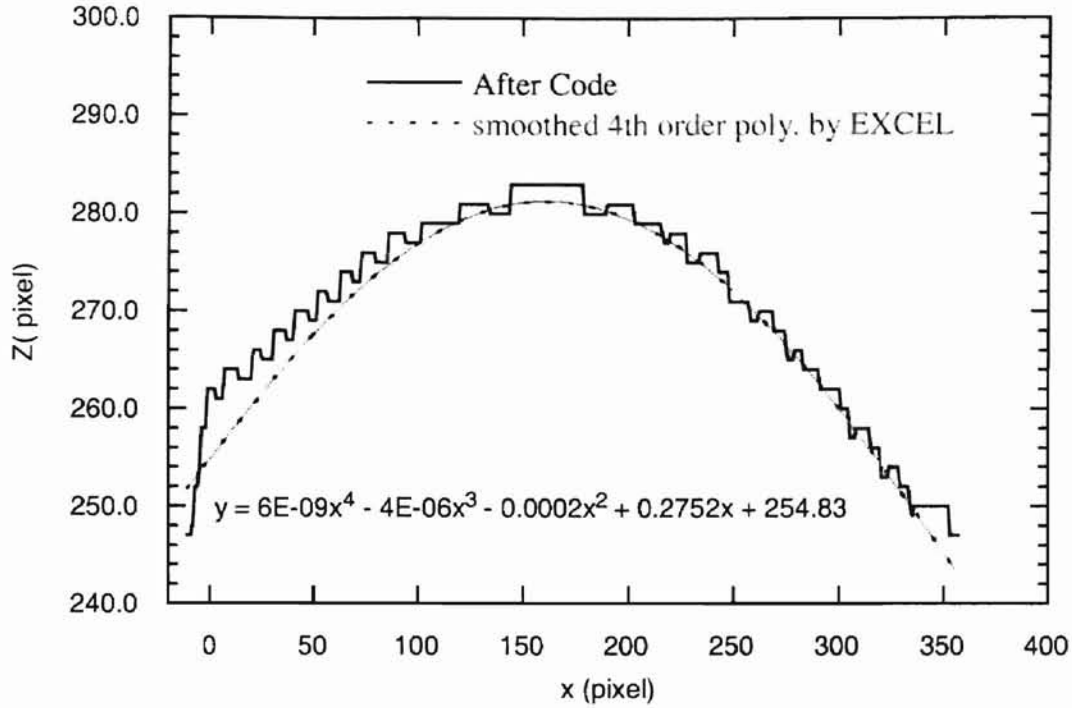

Figure 3-9 4th Order Polynomial Curve Fitting

Figure 3-8 and Figure 3-9 show the curve fitting obtained from Microsoft Excel. However, we find this kind of curve fitting is too coarse to satisfy our need. It is very hard to get just one polynomial math expression to describe the web path.

The problem we need to solve for curve fitting is:

(1) Which point is a reliable web point?

If we look at the output data file, we can see that the gray scale of some web points is too large (>125). This kind of gray scale is not acceptable, and we should eliminate these points.

(2) Which kind of curve fitting method should we use?

It is clear that we must choose a good method to fit that web line. But which function is good? Polynomial, sample interpretation function or else? We must try several.

(3) In general, how large a deviation from the original data is permitted?

If the error of Y is too large, using curve fitting methods to obtain Y' and Y'' will become difficult. We should study the relationship among image resolution, code efficiency and curve-fitting method, and make a balance among them.

I plan to study polynomial first.

Determine the Web Pressure Distribution

In order to determine the pressure distribution underneath the web by using optical methods, we study the web in steady state first.

1) Steady state (simple case):

The relationship among the pressure difference between lower and upper part of the web, the web tension, and the radius of curvature can be derived from the equation of steady motion of the web (Eq.1-1). Suppose the web does not move, so u = 0, and the bending stiffness term EI is negligible compared to other terms. Our equation for the static web is:

$$T \, \partial^2 z / \partial x^2 = P_{gage} \qquad \text{(Eq.3-2)}$$

P_{gage} is local pressure difference, and T is the web tension (lb./in)

A more accurate form is [8]

$$\frac{T}{R} = P_{gage} \qquad \text{(Eq.3-3)}$$

Where R is the radius of the web curvature.

Since:

$$R = \frac{(1+(\frac{dz}{dx})^2)^{\frac{3}{2}}}{d^2z/dx^2} \qquad \text{(Eq.3-4)}$$

So:

$$P_{gage} = T\frac{d^2z/dx^2}{(1+(\frac{dz}{dx})^2)^{\frac{3}{2}}} \qquad \text{(Eq.3-5)}$$

We can measure the web tension T. The role of the camera is to get curvature.

Once we have a series of points on the curve, we can use finite difference expressions, like the first derivative (from 3 points):

$$\left.\frac{dz}{dx}\right|_{x=x_i} = \frac{1}{2}[z(x_i-1)+z(x_i-1)] \qquad \text{(Eq.3-6)}$$

Second derivative (from 3 points):

$$\left.\frac{d^2z}{dx^2}\right|_{x=x_i} = z(x_{i-1})-2z(x_i)+z(x_{i+1}) \qquad \text{(Eq.3-7)}$$

CHAPTER 4

EXPERIMENT

Test Setup

Camera

Tests were performed in order to verify the methods discussed in Chapter 3. In these tests, the digital camera SONY MVC-MD7 was used to obtain the images. The technical specifications of MVC-FD7 are as follows:

Table 4-1 Technical Specifications of Digital Camera

CCD	¼" color CCD, 380K pixels
Image size	640×480
Image Format	JPEG
Recording Media	3.5" 2HD Floppy Disk
Lens	10:1 Optical Zoom, f4.8, 35mm conversion (f = 40-400mm)
Shuttle Speed	1/60- 1/1400 sec

Since the only thing we are interested in is the web edge or centerline, it is not necessary to use "color" mode. The black and white image is enough. The digital camera stands on tripod that can measure the angle of the camera. The resolution mode is "fine" which is the maximum that JPEG compression permits.

Background and Foreground

In order to get good image, it is best to simplify foreground and background. In this test, a white web and black background is used to make the contrast between the web and background maximal. Proper light must be provided.

Air Bar, Web and Measurement Instruments

The air bar used here is an ordinary low wrap "pressure pad" bar. The air flows from slots of both sides of the air bar. The air bar was drilled with 5 holes in the center (CMD direction) to measure the air pressure. Those holes should not affect the aerodynamic forces of the web because they are small compared to air bar.

The web is semi-opaque plastic film with dimensions of 6" × 16". With proper illumination, the web appears as white in camera with dark background.

Since the air supply pressure is not high (the maximum is about 5 inches of water), we use the inches of water to measure the pressure. The air supply pressure, and the pressure of the centerline under the web were measured. The test setup is in Figure 4-1.

Since most digital cameras have some bad CCD pixels, it is desirable to avoid these spots. For our purpose, it is better to let the web be in the lower part of the image. Other effects to be considered are:

(1) distance and angle between camera and object

(2) illumination

(3) web deformation

(4) web tension

After the proper adjustment, the image is like Figure 4-2.

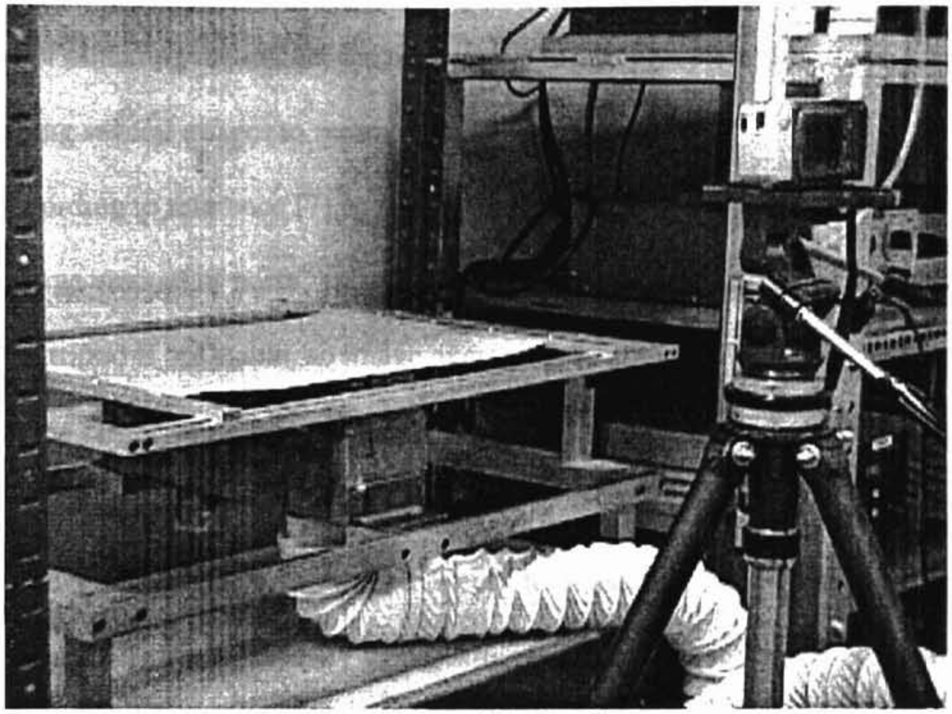

Figure 4-1 Test Setup

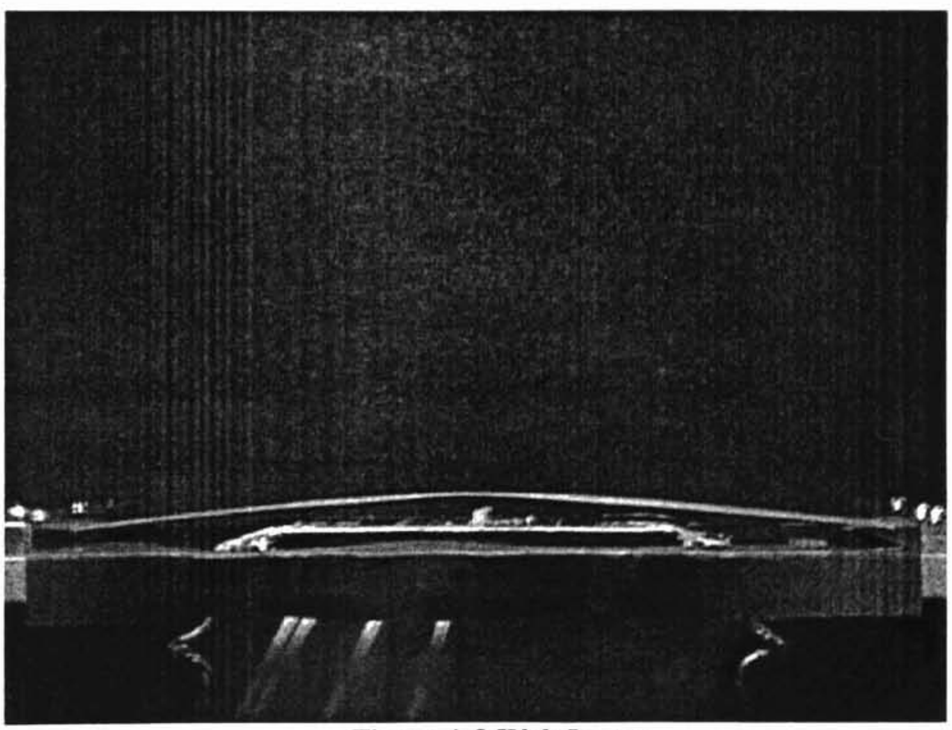

Figure 4-2 Web Image

Image Processing

After obtaining the image, the next step is image processing:

(1) Averaging several images to remove the noises

According to reference [7], it is enough to average 3-4 images, "but even 2 will remove a large amount of noise" (p.84). In our case, no good sharp edge was obtained because the tripod is not stable, so image averaging. is not used.

(2) Detect the center white line and using Center of Gravity Method (CGM) to reach sub pixel resolution

Since the white line is the web, it is easy to detect that line by using the computer code to find the maximum intensity value. After finding the first web point, the rest of the white points are obtained by comparing the intensity value of pixels in neighborhood. By using the code "centernew.c", it is easy to get the relatively smoothed web line compared with Brigitte Bush's edge-detection method. The detailed results will be shown in the next chapter.

The steps are indicated in Figure 3-1. The code is attached in the appendix.

The Results of Measurement

In order to compare the pressure obtained from the experiment with the results of the image processing method, we need to measure the pressure under the web. The 12 test points are indicated in the following graph:

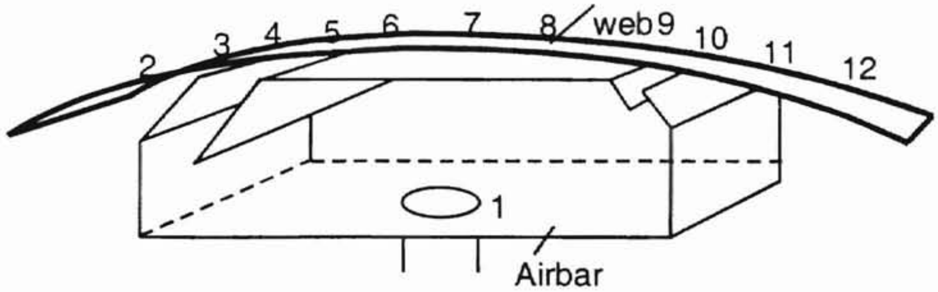

Figure 4-3 The Point of Measurement

In this test, the maximum flotation height of the web (approximately point #6) is from 0.35 to 1.5 inches, and the web length (between two rods) is from 9 to 10 inches. Because the shapes of pressure profiles are similar, only one case is listed here to illustrate the problem. The values of each point's pressure are listed in the following table when the web flotation height is 0.35.

Table 4-2 The Test Points' Pressure

#	#1	#2	#3	#4	#5	#6
X(in)	0"	-4"	-3"	$-2^{13/16}$"	-2"	-1"
Pressure(inch of water)	4.24	-*	-*	3.2	1.88	1.94

#	#7	#8	#9	#10	#11	#12
X(in)	0"	1"	2"	$2^{3/16}$"	3"	4"
Pressure(inch of water)	2	1.96	1.9	3	-*	-*

(*: The values of these cells are minus)

The shape of the pressure profile for the above configuration is in the following figure:

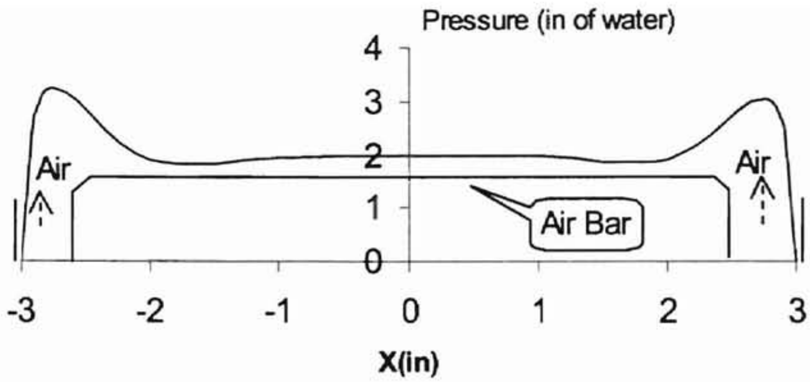

Figure 4-4 Pressure Distribution of the Web

CHAPTER 5

RESULTS

Web Line Detection

The web image is like the following figure:

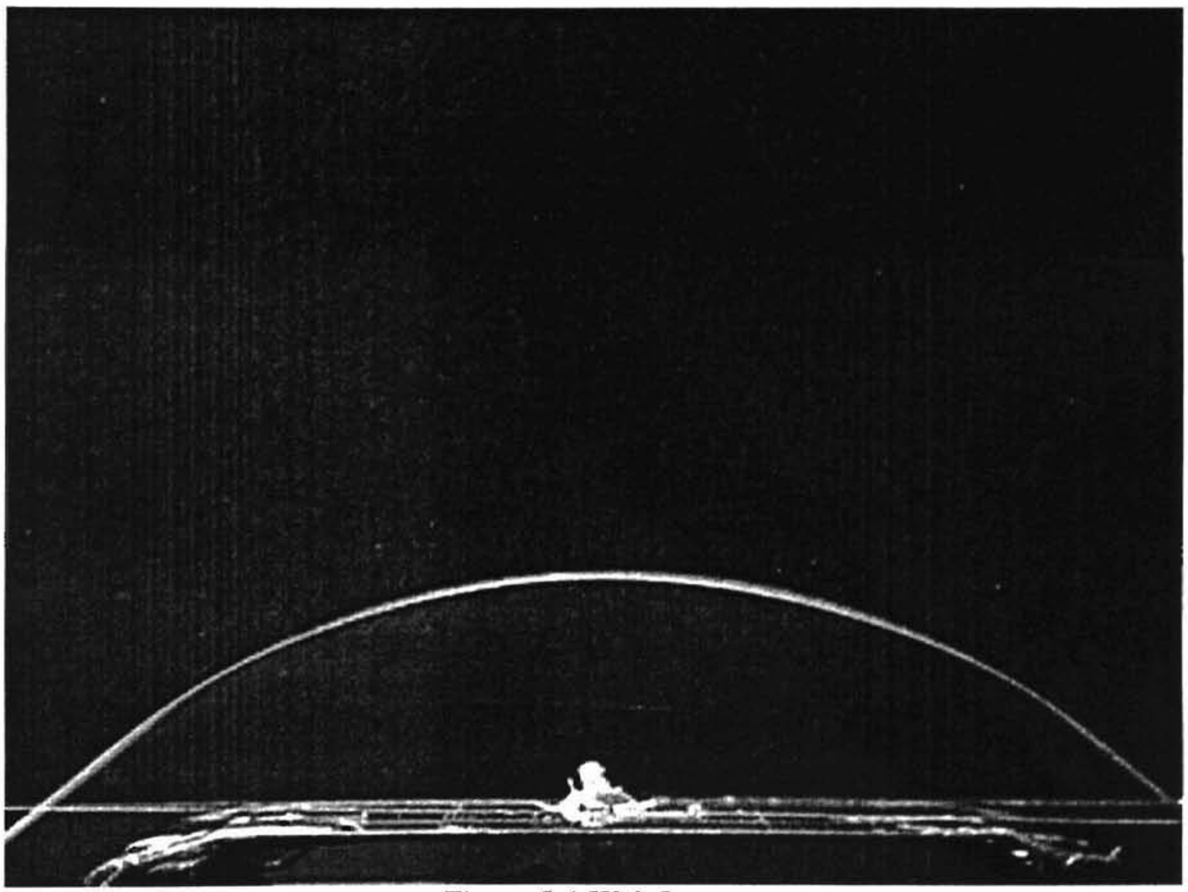

Figure 5-1 Web Image

It is noticed that the web background is dark and the web is relatively white. This makes it easier for the code to detect the web line. In this code, the search range for the first web point is from the top of the image. In most cases, it is more than 240 pixels. After the first web point was detected, the search range can be small, since the web line is continuous and it is not necessary to search a large range. Figure 5-2 is the web line obtained by detecting the brightest points in image.

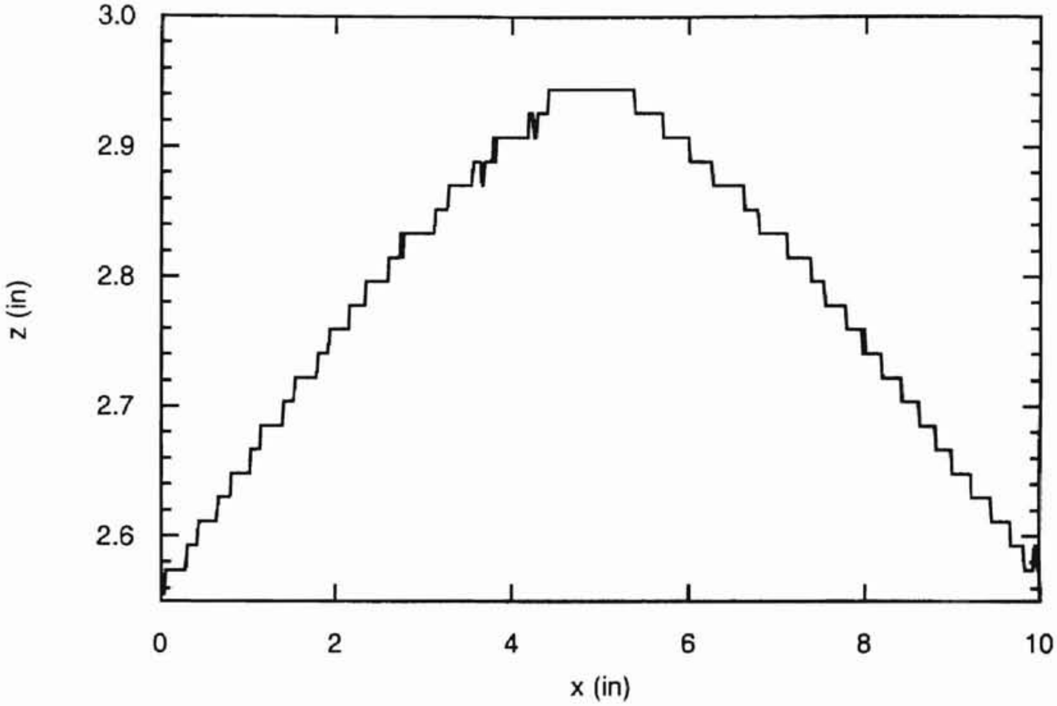

Figure 5-2 Web Line

From the above figure, it is observed that the line is not so smooth since there are large steps in it. After using CGM method, the line looks like Figure 5-3.

From Figure 5-3, we can see the line is smoother than that in Figure 5-2. This is because the information from the neighborhood of the web line is used, and the whole image's resolution in Z direction is increased to sub-pixel level.

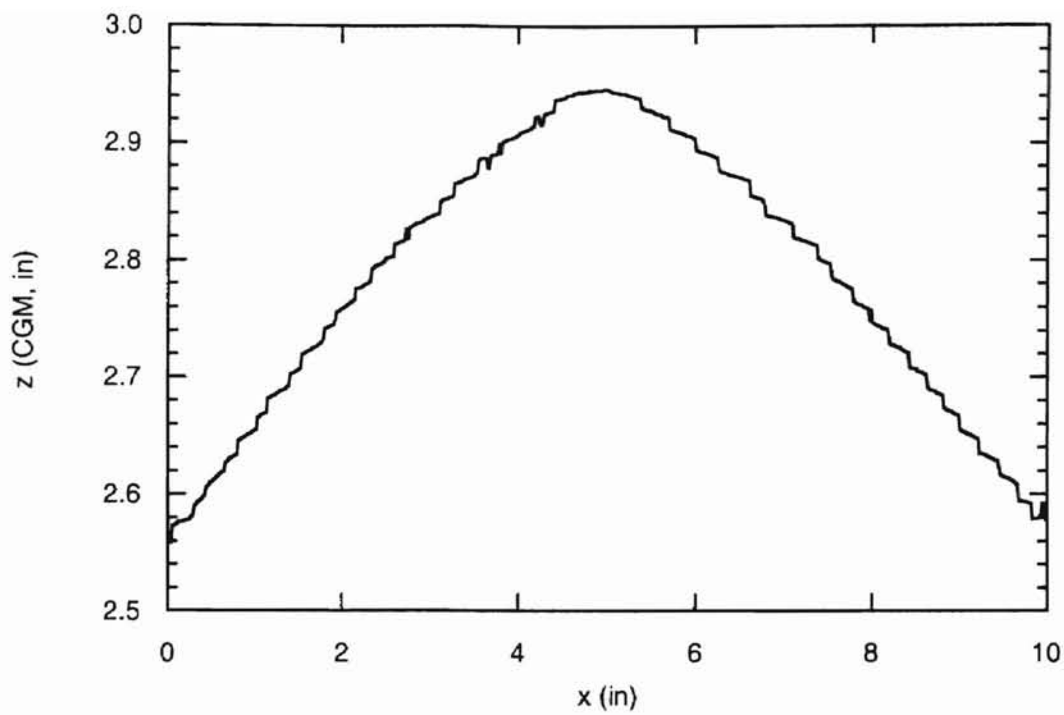

Figure 5-3 Smoothed Web Line (After CGM method)

Curve Fitting

After getting the web line, we use curve-fitting methods to find the model that can represent the problem.

There are several possible models for the problem:

Polynomial Curve Fitting

Since every complicated math expression can be expanded in polynomial, it is a good idea to try a polynomial model first.

One of polynomial curve fitting models is:

$$z(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n = \sum_{i=1}^{n} a_i x^i \qquad \text{(Eq. 5-1)}$$

The order n depends on the real situation. If we increase the order n, the computational time increases. If we do not, the result is not satisfactory. We must trade off between simplicity and accuracy.

Curve Plot and SAS are employed to fit the data point. Some of the results are as follows:

n=5:

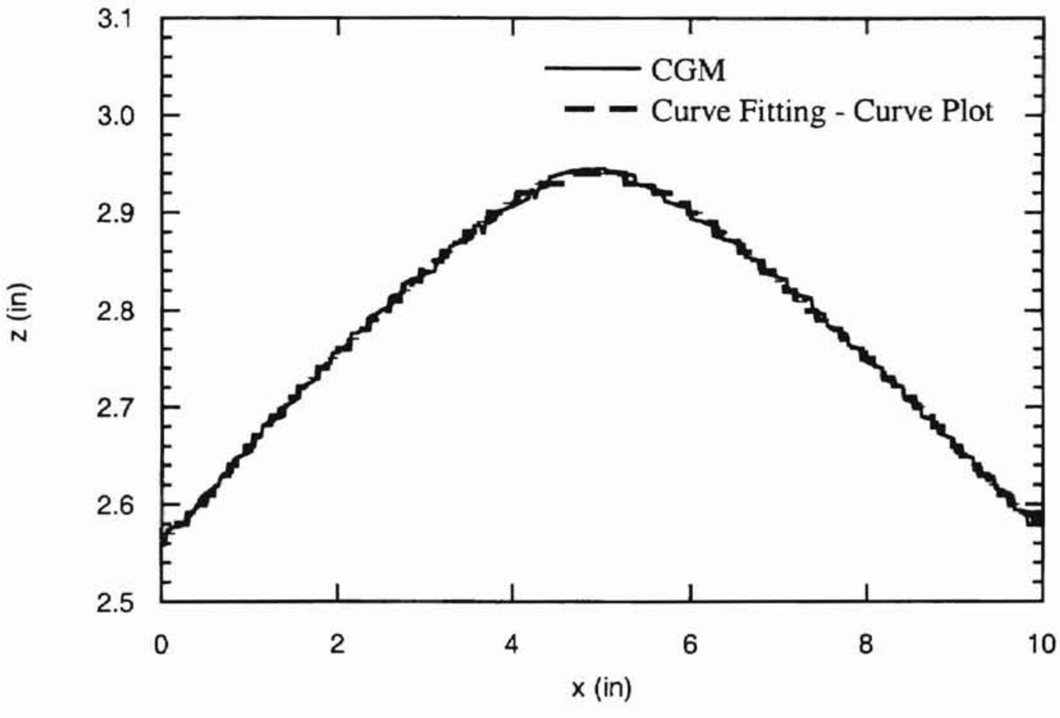

Figure 5-4 Curve Fitting by 5th Order Polynomial

From the above figure, we can see that in the center area, 5th order polynomial can not fit the data points very well.

If we increase the order of polynomial, we can get the following curve fitting results (n=9):

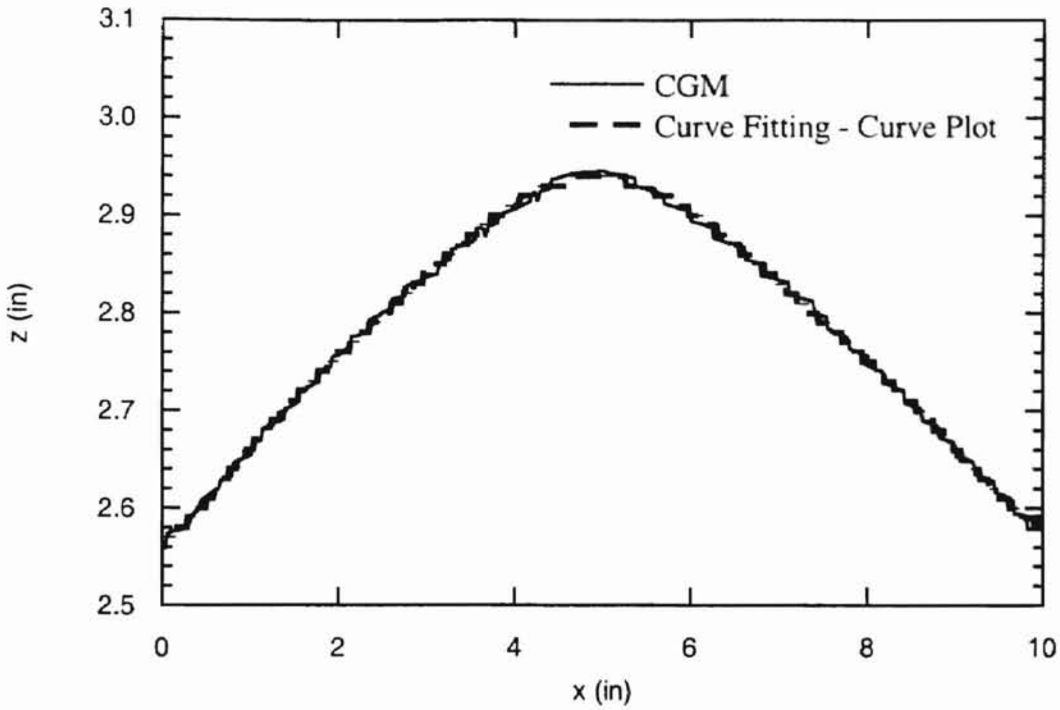

Figure 5-5 Curve Fitting by 9th Order Polynomial

It is clear that 9th order is better than 5th order polynomial in the center area. If we consider the web and air bar as almost symmetric, we can use even polynomial expressions to fit the data:

$$z(x) = a_0 + a_1 x^2 + a_2 x^4 + ... + a_n x^{2n} = \sum_{i=1}^{n} a_i x^{2i} \tag{5-2}$$

If n=4, the curve fitting result is:

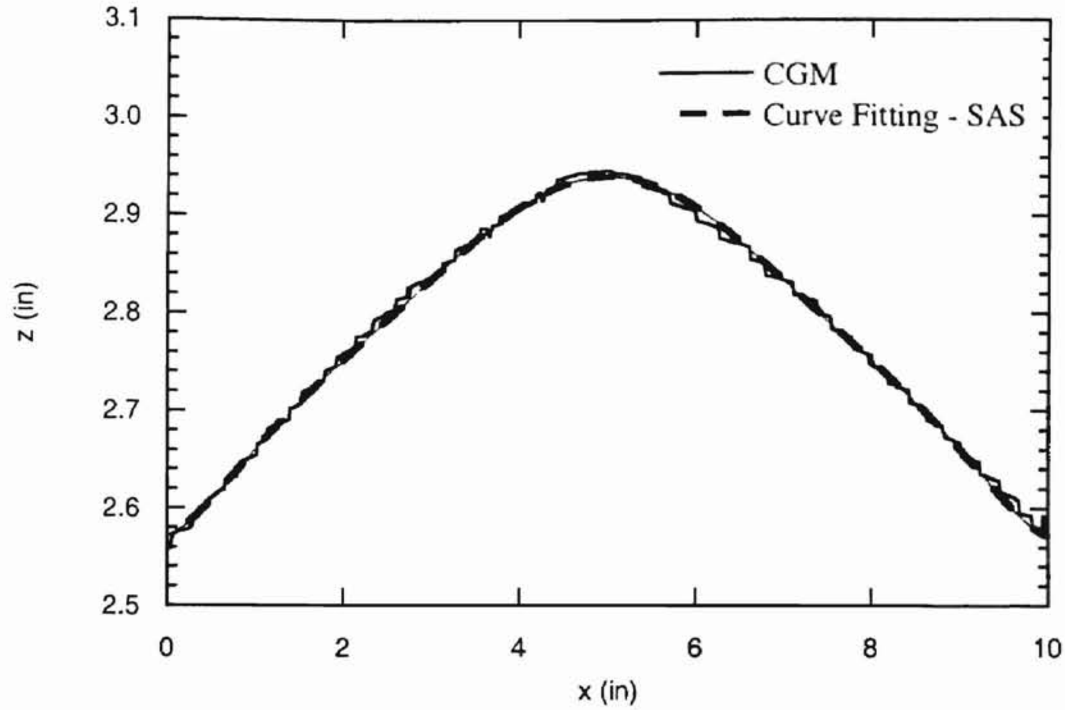

Figure 5-6 Curve Fitting by 8^{th} Order Polynomial (n=4)

Sinusoidal Curve Fitting

The idea of sinusoidal curve fitting is from Fourier analysis. Periodic excitations can be represented by the sum of a series of sinusoidal functions. If we treat the web in the test as a near sinusoidal waveform, we can fit the web with sinusoidal functions.

According to the above idea, the web length L should be half of the wavelength λ.

So: $$\lambda = 2L \qquad \text{(Eq. 5-3)}$$

Where: L is the length of the web.

The sinusoidal function has the form:

$$z(x) = a_0 + a_1 \sin(\frac{2\pi}{\lambda}x) + a_2 \sin(\frac{4\pi}{\lambda}x) + ...a_n \sin(\frac{2n\pi}{\lambda}x) + b_1 \cos(\frac{2\pi}{\lambda}x) + ... + b_n \cos(\frac{2n\pi}{\lambda}x)$$

Or: $$z(x) = a_0 + \sum_{i=1}^{n} a_i \sin(\frac{2i\pi}{\lambda}x) + \sum_{i=1}^{n} b_i \cos(\frac{2i\pi}{\lambda}x) \qquad \text{(Eq. 5-4)}$$

If we try sinusoidal function, we can find the following interesting curve-fitting results (n =3):

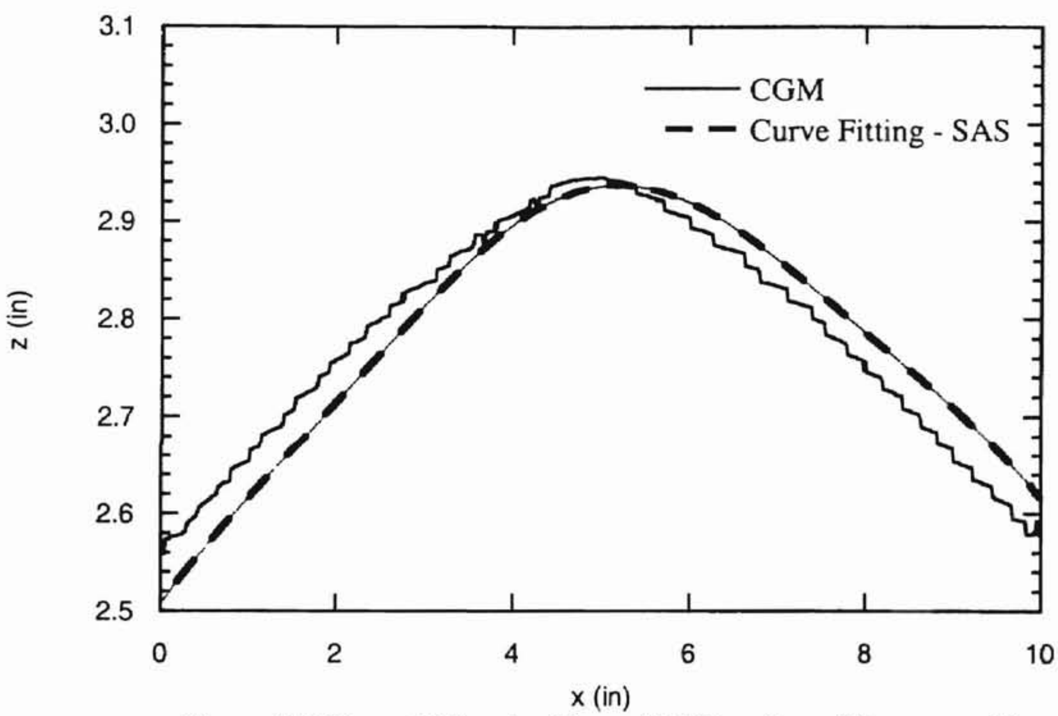

Figure 5-7 Curve Fitting by Sinusoidal Functions (7 terms, n=3)

If given n = 4, the result is as Figure 5-8.

From Figure 5-7 and Figure 5-8, we can see that 9 terms of sinusoidal function is better than 7 terms. If given n = 5, result as Figure 5-9 can be obtained.

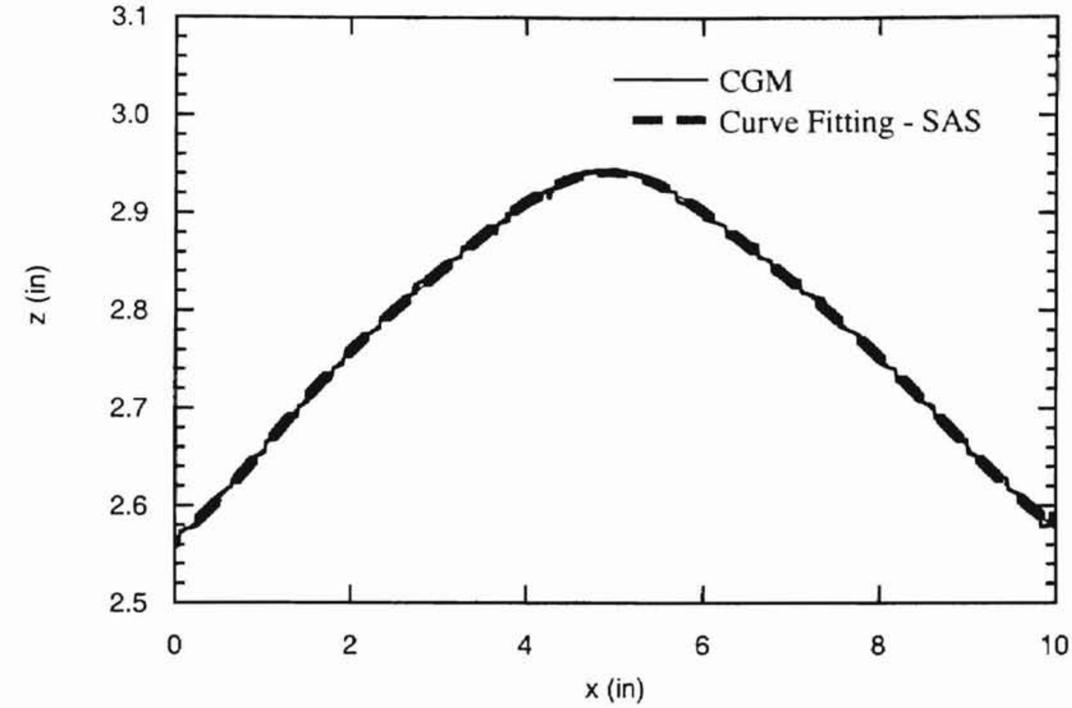

Figure 5-8 Curve Fitting by Sinusoidal Functions (9 terms, n=4)

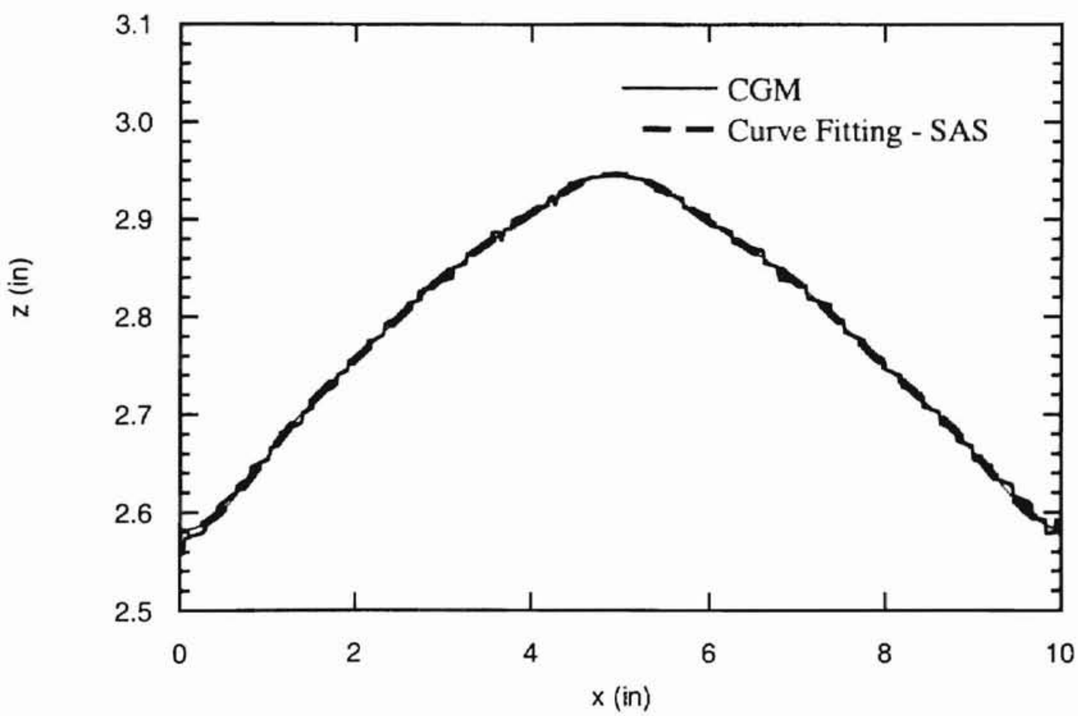

Figure 5-9 Curve Fitting by Sinusoidal Functions (11 terms, n=5)

We almost can not see the difference between Figure 5-8 and Figure 5-9. From the SAS's Analysis of Variance, the comparison between them is as follows:

Table 5-1 Comparison between Figure 5-8 and Figure 5-9

Parameters	Root MSE	Dep Mean	C.V.	R-square	Adj R-sq
n=4(Figure 5-7)	0.00419	2.78317	0.15065	0.9987	0.9987
n=5(Figure 5-8)	0.0038	2.78317	0.13638	0.9989	0.9989

From the above table, we can see that n=5 (Figure 5-9) is better than n=4 (Figure 5-8). If we try different λ, we can also get quite good results:

λ=10(n=11):

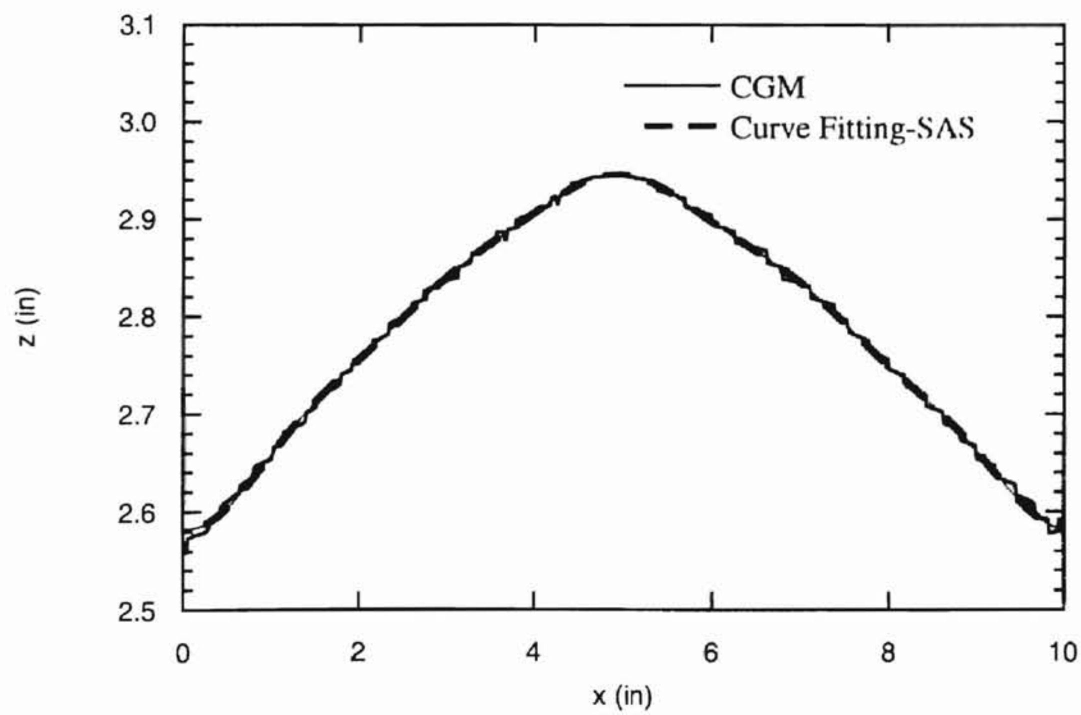

Figure 5-10 Curve Fitting by Sinusoidal Functions (λ=10, 11 terms, n=5)

If we use a different λ, for example, λ = 12 inches, similar results can be obtained.

Web Pressure Profile

When we acquire the web deflection, we can develop the pressure profile of the web. According to Eq. 3-2, the air pressure on the web is:

$$P_{gage} = T\frac{d^2z}{dx^2} \qquad \text{(Eq. 3-2)}$$

In the above equation, the second derivative is the essential factor to find P_{gage}. Since the web tension is not known, we can not derive the pressure profile of the whole web. But we can measure the pressure at center point. By re-scaling, the tension T can be determined so that the pressure P_{gage} at center point is equal to the measurement. Different curve-fitting models will produce different pressure profiles.

Pressure Profile by Polynomial and Sinusoidal Curve Fitting

After the curve fitting, the first and second derivatives can be calculated from the curve math expression. Figure 5-11 is one of the results (h=1.5 inches, L=9 inches). From Figure 5-11, it is noticed that the measurement is similar to curve fitting results. The reason why 9^{th} order polynomial is chosen is that: the curve of measurement can be treated as 6^{th} order polynomial (after my test, 6^{th} order polynomial can fit measurement quite well). After this polynomial is double integrated, it will become a 8^{th} order polynomial. Considering the error terms, 9^{th} order polynomial is a reasonable choice.

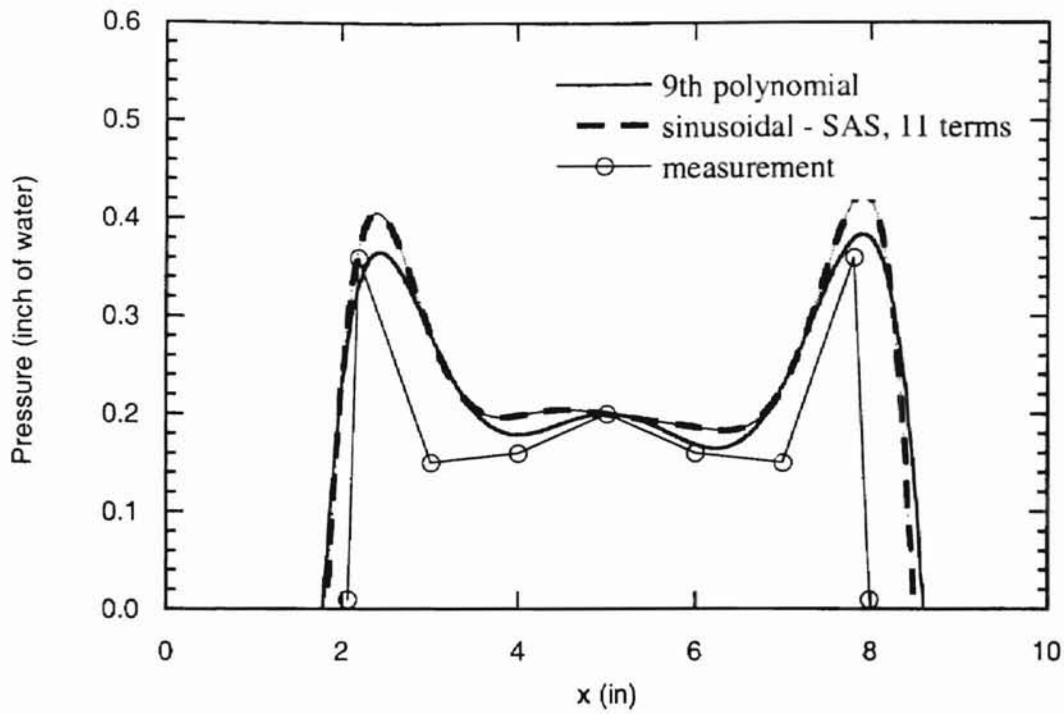

Figure 5-11 Pressure Profile from Different Curve-Fitting Methods

Pressure Profile by Different Flotation Height

h = 1.5 inches, L = 9 inches: Since L = 9 inches, λ should be the double of L, so λ = 18 inches. Polynomial and sinusoidal curve fitting methods are tried, and the pressure profile is plotted in Figure 5-11.

h = 0.86 inch, L = 9 inches: For λ=18 inches, the pressure profile is plotted in Figure 5-12.

h=0.35 inch, L=10 inches: For λ=20 inches, several different n, from 5 to 11, were tried. The pressure profile is plotted in Figure 5-13. It is noticed that the analytical results have large differences from measurement. A possible reason is that the vertical resolution of the digital camera is not high enough. From the image taken, we can see the vertical pixels that the web spans are less than 20. The whole web line is quite "flat". After the CGM method, although the resolution in vertical direction can be increased to 40-70 for web, it is still too small compared with horizontal pixels the web occupies. In the next chapter, there are discussions about how to improve it.

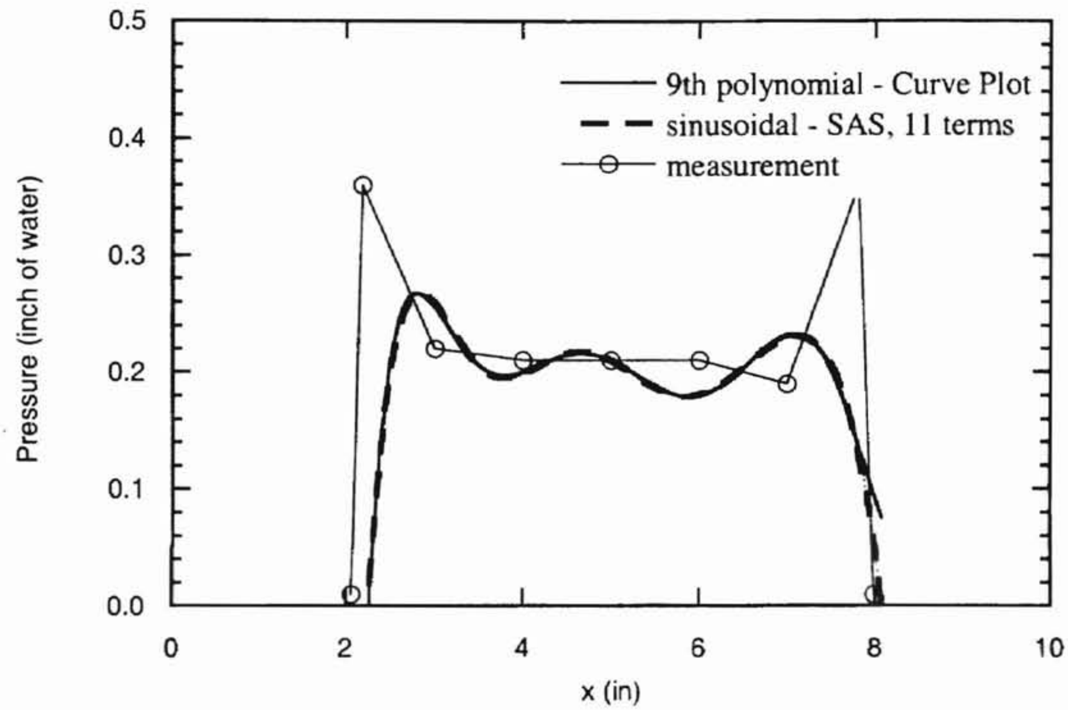

Figure 5-12 Pressure Profile (h=0.86 inch)

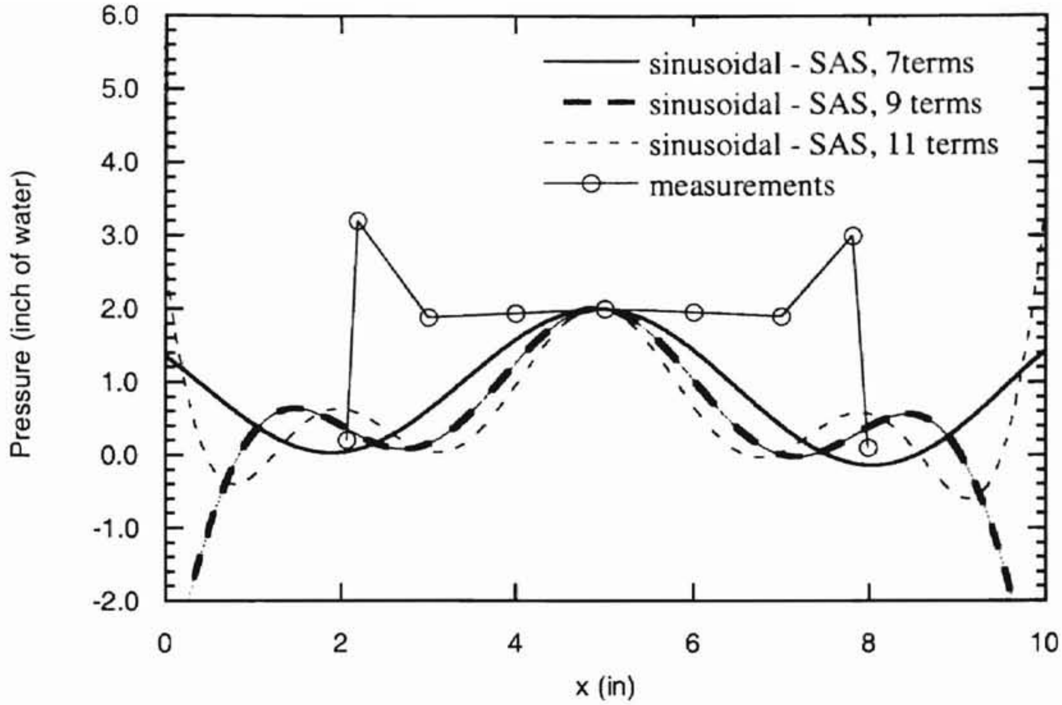

Figure 5-13 Pressure Profile (h=0.35 inch)

CHAPTER 6

CONCLUSIONS AND DISCUSSION

Conclusions

The intent of this study is to determine the out-of-plane displacement of the web and to obtain the pressure profile. The investigation has shown that image processing method can be used to determine the web displacement, provided that the displacement is not too small. Using digital camera, the image's resolution can be increased over the previously used framegrabber. By using CGM method (Eq. 3-1), sub-pixel image resolution in the vertical direction can be achieved. By using proper curve fitting method, the pressure profile can be obtained by Eq. 3-2:

$$T \frac{\partial^2 z}{\partial x^2} = P_{gage} \qquad \text{(Eq. 3-2)}$$

Conclusion 1: The shape of the pressure distribution profile obtained from optical deflection is similar to the measured values for a flotation height from 0.8 to 1.5 inches.

For different web flotation heights h, it is observed that the result is good if h > 0.8 inch. The pressure profiles from both measurement and image processing analysis show the maximum air pressure of the web occurs at the outlet of air bar's nozzle. Also the results show that the center pressure of air bar is a little bigger than its neighborhood.

Conclusion 2: The image resolution is decisive for the method's success.

The success of the image processing method is determined by four factors: (1) the resolution of the digital camera; (2) the number of vertical pixels the web occupies; (3) the contrast between background and foreground; (4) the proper curve-fitting method. Among these 4 factors, (1), (2) and (3) are the matter of image resolution, especially the vertical resolution. It is shown that the vertical resolution can reach 1/3 to1/5 pixel by proper setup and using the CGM method.

Conclusion 3: Proper curve fitting is vital.

By using a proper model, the curve fitting method can lead to good results. For polynomial curve fitting, the order should be greater than 8. For sinusoidal curve fitting, the wavelength should be the half distance between the web's two fixed points.

Curve fitting can be done by SAS, Table Curve 2D or Curve Plot, etc. The curve fitting method is vital in obtaining results.

In this thesis Curve Plot is used to fit the measurement. The model is polynomial. After trial, it is found the 6^{th} order of polynomial can fit the measured data well. By using SAS or Curve table 2D, different models can be developed for fitting data obtained from the digital camera. For example, one of the models is based on the property that airbar is symmetric about its middle axis. The curve-fitting model can be built as follows:

$$z(x) = a0 + a1 \times (x-5)^2 + a2 \times (x-5)^4 + a3 \times (x-5)^6 + a4 \times (x-5)^8 \qquad \text{(Eq. 6-1)}$$

Where: $x = 5$ is the middle axis of air bar

By using SAS, this symmetric configuration can achieve a reasonable result.

Discussion

Vertical Resolution

From the results (Figure 5-11, Figure 5-12), we can see the image's resolution is still the main factor that influences the success of the optical method. In this project, the CGM method is used to increase the vertical resolution. However, when deflection is small, this method is still not accurate enough to generate a reasonable second derivative. Thus, the use of cylindrical lens may be helpful.

Cylindrical Lens: A cylindrical lens magnifies one dimension only. By mounting the cylindrical lens in front of the camera, a vertical resolution may be obtained. Figure 6-1 and Figure 6-2 show how this cylindrical lens can magnify vertical direction.

From Figure 6-1 and Figure 6-2, we can see the vertical pixels of the web are increased. However, the image is not so clear. With current criteria, it is difficult for our code to distinguish which is the web line. So for this kind of image, a new code needs to be developed.

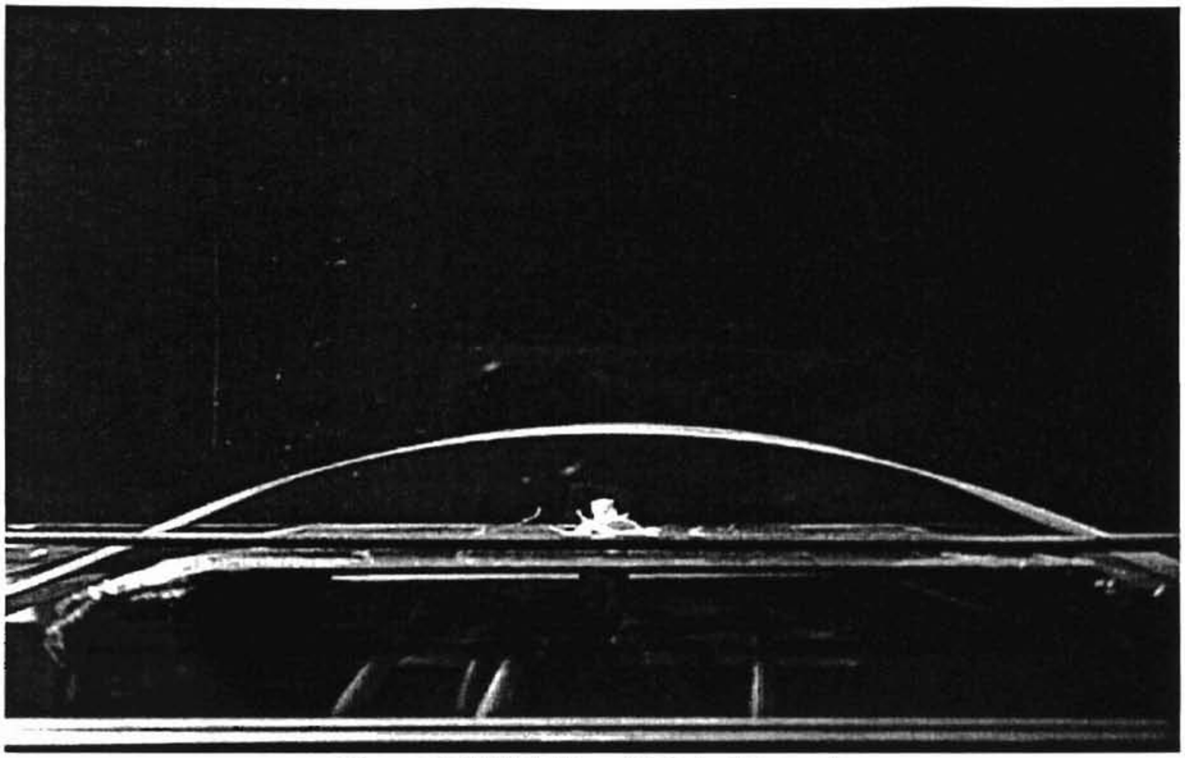

Figure 6-1 Web Line (Original Image)

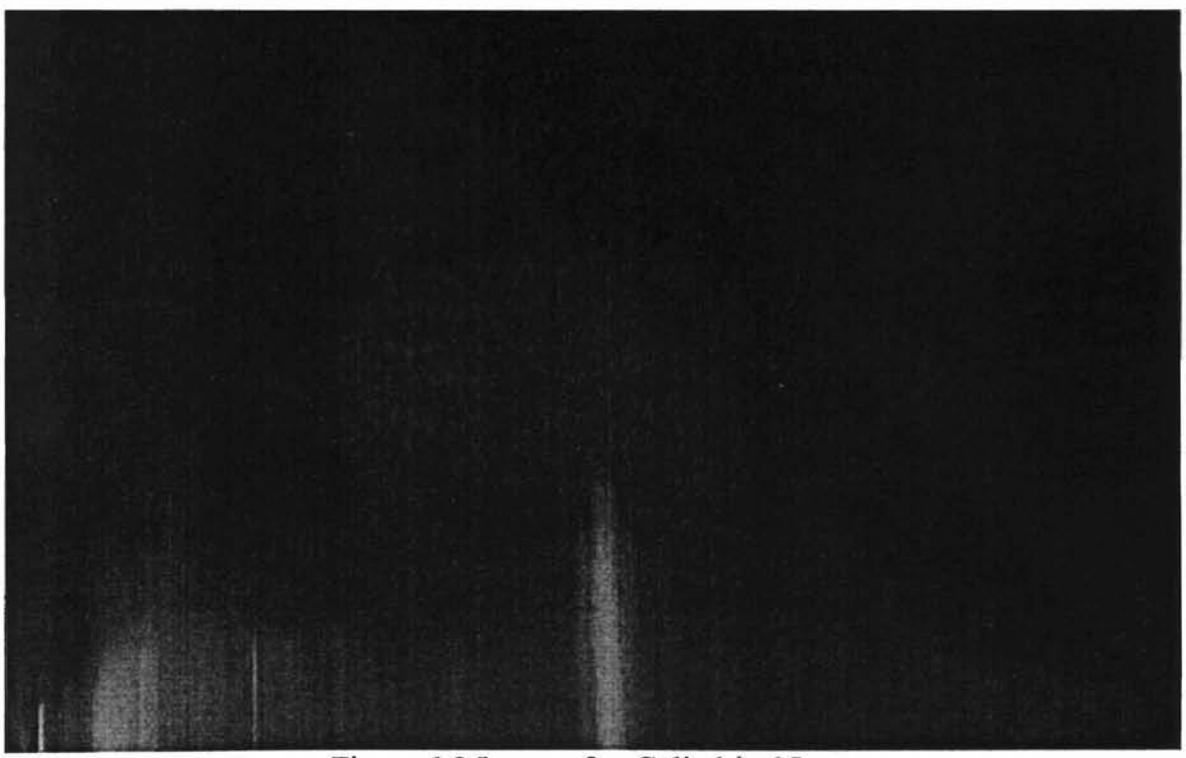

Figure 6-2 Image after Cylindrical Lens

Cylindrical Lens + Concave Lens: Since the above image is not expected for the code to detect the web line, it is necessary to consider concave lens to reduce the focus problem. After installing the concave lens before the cylindrical lens, the test image is shown in Figure 6-3 and Figure 6-4.

It is observed that the image resolution after concave lens and cylindrical lens improves a little bit. Although much effort has been made to improve image quality, it is still difficult for image processing code to detect the whole line. In a word, there is a focus problem in camera system: the center is more clear compared with the other part of the line. It is almost impossible to detect the line except the middle part. Probably it needs a more robust code.

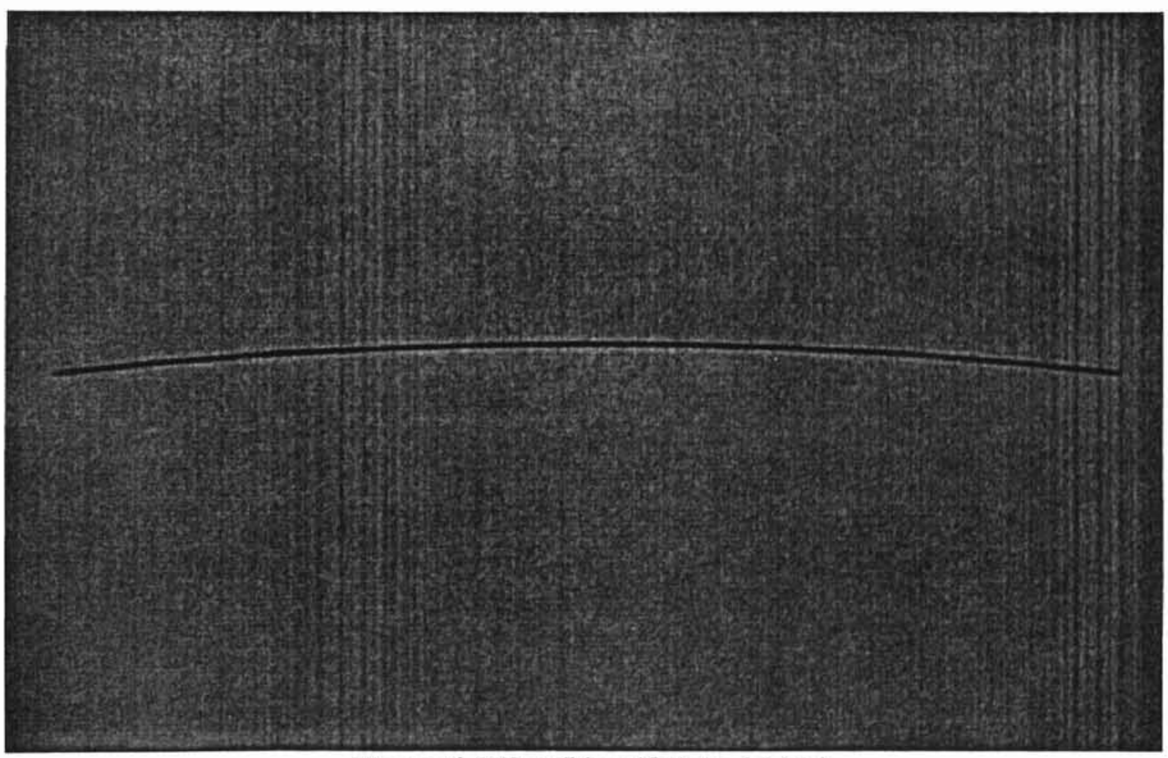

Figure 6-3 Test Line (Original Line)

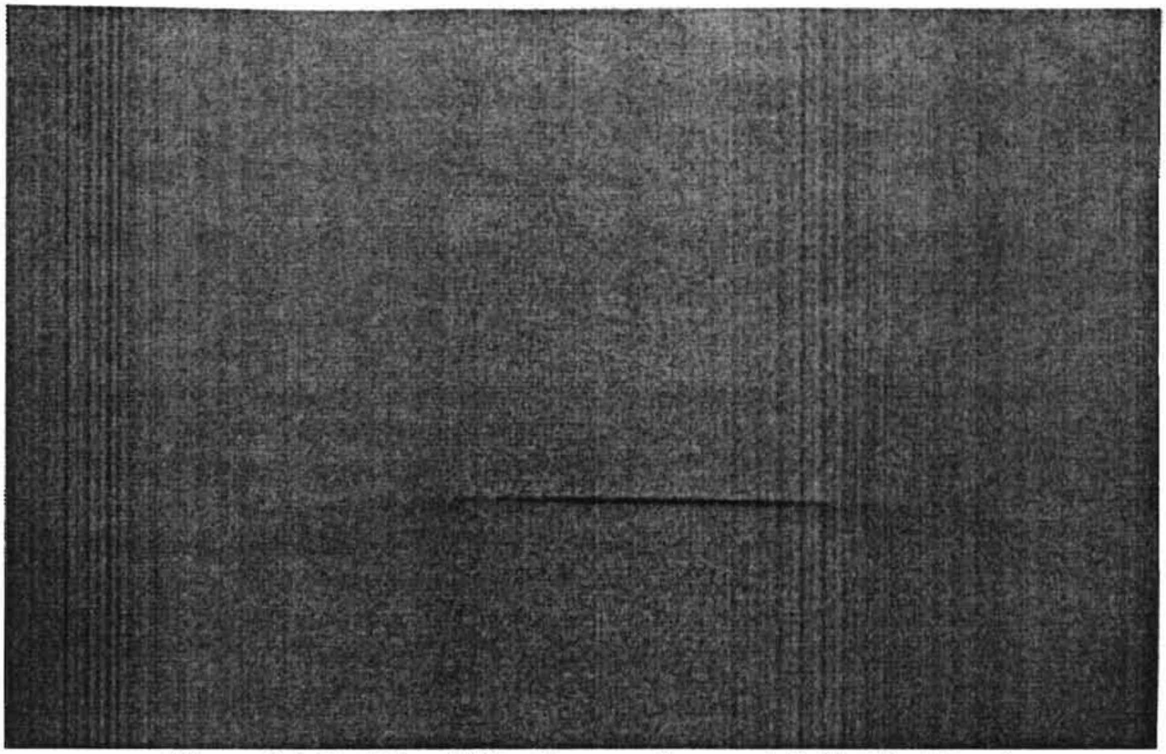

Figure 6-4 Test Line (After Concave Lens and Cylindrical Lens)

Curve Fitting Model

The proper curve-fitting model is an important factor that can influence the result significantly. There is a possible way to get better models.

This method is to obtain a model that may be derived from the literature [11], in which the ground effect theory is used to model the pressure profile of the air bar. The resulting equation could be used as a model for the curve to be fitted.

REFERENCES

[1] Abbott, A. L., et al., "Finding Lines and Building Pyramids With Splash2", Proceedings of IEEE Workshop on FPGA's for Custom Computing Machines (0-8186-5490-2/94), 1994 IEEE, pp.155-163, 1994.

[2] Anson, S.I, "Identification of Periodic Marks in Paper and Board by Image Analysis Using Two-Dimensional Fast Fourier Transforms", Tappi Journal, Vol. 78, No. 3 pp.113-119, 1995.

[3] Behar, D., Cheung, J., and Kurz, L., "Contrast Techniques for Line Detection in a Correlated Noise Environment", IEEE Transactions on Image Processing (1057-7149/97) Vol. 6, pp.625-641, May 1997.

[4] Berndtson, J. and Niemi, A.J., "Automatic Observation of the Dry Line in Paper Machine", IEEE Proceedings Of ICPR '96, pp.308-312, 1996.

[5] Bolotin, V.V., "Nonconservative Problems of the Theory of Elastic Stability", The Macmillan Company, 1963.

[6] Budd, G.W., "Machine Vision Application Using Binary and Grayscale Morphology", Tappi Journal, Vol. 79, No. 8, pp. 103-111, August 1996.

[7] Burdick, H.E., "Digital Imaging Theory and Applications", McGraw-Hill

[8] Busch, B., "Measurements on Air Bar/Web Interaction for the Determination of Stability of a Web", MS. Thesis, Oklahoma State University, 1996.

[9] Chan T.S. and Raymond, K.K.Y., " Line Detection Algorithm", Proceedings of ICPR '96, pp.126-130, 1993.

[10] Chang, Y.B., "An Experiment and Analytical Study of Web Flutter", Ph.D. Dissertation, Oklahoma State University, 1991.

[11] Chang, Y.B. and Moretti, P.M, "Aerodynamic Characteristics of Pressure-Pad Air Bars", Proceedings of The 1997 ASME Winter International Conference, Dallas, Texas, pp.748-751, October 17-21, 1997.

[12] Dehili, A. and Akil, M., et al., "Parallel Hough Transform on a Hierarchical Structure", Proceedings of ICPR '96, pp.522-526, 1996.

[13] DeWitte, J.A., William L.B. and Scott B. P., "Video Surveillance Troubleshooting at LSPI", Tappi Journal, Vol. 78, No. 8, pp.103-110, March 1995.

[14] Dowell, E.H., "A Modern Course in Aeroelasticity", Kluwer Academic Publishers, 1995.

[15] Grennhill, D. and Davies, E.R., " A New Approach to the Determination of Unbiased Thresholds for Image Segmentation", Fifth International Conference on Image Proceeding and Its Applications, pp.519-523, 4-6 July 1995.

[16] Guo, J. and Xu, G., " Precision estimation of a Fitted Image Line", IEEE International Conference on IEEE System Engineering (CH2767-2/89/0000-0023) 1989 IEEE, pp. 23-26,1989.

[17] Hendry, A., Skingley, J. and Rye, A. J., "Automated Linear Detection and its Application to Curve Location in Synthetic Aperture Radar Imagery", Proceedings of IGARSS '88 Symposium, pp.1521-1524, Edinburge, 13-16 September 1988.

[18] Hinteregger, H. and Muftu, S., "Contact Tape Recording with a Flat Head Contour", IEEE Transactions on Magnetics, Vol. 32, No. 5, pp.3476-3478, September 1996.

[19] Keren, D., Cooper, D., and Subrahmonia, J., "Describing Complicated Objects by Implicit Polynomials", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 16, No. 1, pp.38-52, January 1994.

[20] Kodak Co.'s Handout, "Air Conveyance", 1997.

[21] Lee, C.K., Lee, T.C., and Ho Y.W., "Non-Contact Strain Measurement", IEE International Conference on Image Processing and its Applications, pp.176-180, 1992.

[22] Liang, P.," A New Transform for Curve Detection", Proceedings of the Third International Conference of Computer Vision (CH2934-8/90/0000/0748 1990 IEEE), Osaka, Japan, pp.748-751, 1990.

[23] Lim, G. and Alder, M., "A Non-Parametric Approach for Detecting Lines and Curves", Proceedings of 3rd IEEE International Conference on Image Proceeding, Vol. 1, pp.837-840, Sept. 16-19, 1996.

[24] Merter, N. and Zerubia, J., "New Prospects in Line Detection by Dynamic Programming", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 4, pp.426-431, April 1996.

[25] Moretti, P.M., "Out-of Plane Dynamics of a Moving Web (8800-2)", WHRC Project Report, pp.18-19, October 1997.

[26] Muftu, S. and Benson, R.C., "Numerical Simulation of Tape Dynamics in Helical-Scan Recording", IEEE Transactions On Magnetics, Vol. 29, No. 6, pp.3927-3929, November 1993.

[27] Nordstrom, A., Carlsson, L.A. and Hagglund, Jan-Erik, " Measuring Curl of Thin Papers", Tappi Journal, Vol. 80, No. 1, pp.238-244, January 1997.

[28] Palmer, P. L., Kittler J., and Petrou M., "A Hough Transform Algorithm with a 2D Hypothesis Testing Kernel", 11th IAPR International Conference on Pattern Recognition (0-8186-2920-7/92), 1992 IEEE, pp.276-279, 1992.

[29] Peincen J., Illingworth J. and Kittler J., "Hypothesis Testing: a Framework for Analysing and Optimising Hough Transform Performance", Third International Conference on Computer Vision (CH2934-8/90/0000/0427), 1990 IEEE Comput. Soc. Press, pp.427-434, 1990.

[30] Sugawara, K., "Weighted Hough Transform on a Gridded Image Plan", Proceedings of The Forth International Conference on Document Analysis and Recognition (0-8186-7898-4/97 1997 IEEE), Ulm, Germany, pp.701-704, 1997.

[31] Verbeek, P.W., Van V., and Lucas J., "Line Detection by Symmetry Filters", IEEE 11th IAPR International Conference on Pattern Recognition (0-8186-2920-7/92), Vol. 3, pp.749-753,1992.

[32] Wolfgang, B., "Line Detection in Discrete Scale-Space", IJCNN-91-Seattle International Joint Conference on Neural Networks (0-8186-6950-0/94), IEEE, pp.915-918, 1994.

[33] Yang, M.C.K., et al., " Hough Transform Modified by Line Connectivity and Line Thickness", IEEE Transactions on Pattern Analysis and Machine Intelligence (0162-8828/97), Vol.19, pp. 905-910, August, 1997.

Appendix

Source C Code for Image Processing

```
/*                    Name:    centerbk.c
 *                    Author: WANG, Kai & FANG, Xiangming
 *                    Date:    Dec. 1998

 * Synopsis - This code reads in an image file (*.raw) with 640*480
   pixels resolution and outputs several lines of black points into a
   file. The format to execute it is "centerbk.exe <input file>
   <output file>".

   /*************************** main() ********************************/
   /* Parameters:
      1) argc - number of command line parameters, should be 3 in this
                program
      2) argv - command line parameters

    * Variables:
      1) fileout - file to hold data output
      2) image - address of a two dimensional array to store image
          intensity
      3) points - array to store black points of the central/edge web
         line
      4) morepoints - two dimensional array to store black points of
         other web lines
      5) linendx - scope of web lines
 */

#include <stdio.h>
#include <stdlib.h>

#define ROWS    480                 /* rows in image file (*.raw) */
#define COLUMNS 640                 /* columns in image file (*.raw) */

typedef struct {
      int x;
      int y;
      int ints;
```

```
      }pointtype;

struct center {
  int x;
  float y;
  };

/* Declare Prototypes */
unsigned char **read_image ( char *filename );
void find_point ( pointtype point[], unsigned char **data );
void find_more_line ( pointtype morepoint[][COLUMNS], int linenum,
                      pointtype point[], unsigned char **data );
void change_coord ( pointtype point[], pointtype morepoint[][COLUMNS],
                    int linenum );
void center_of_gravity ( struct center centerpoint[],
                         pointtype point[],
                         pointtype morepoint[][COLUMNS], int linenum );
FILE *write_image ( char *filename, pointtype point[],
                    pointtype morepoint[][COLUMNS], int linenum,
                    struct center centerpoint[] );

void main ( int argc, char *argv[] )
{
      FILE *fileout;
      unsigned char **image;
      pointtype points[COLUMNS], morepoints[10][COLUMNS];
  struct center centerpoints[COLUMNS];
  int linendx;

  /* check if the command line parameters are correct */
      if ( argc != 3 )
      {
            printf ( "Usage: centerbk.exe <inputfile>
                        <outputfile>\n" );
            exit ( 1 );
      }

  printf ( "Please input the scope of lines (must be less than 5): " );
  scanf ( "%d", &linendx );

  /* Step 1: Read image to array image[][]. */
      image = read_image ( argv[1] );

  /* Step 2: Find web line. */
      find_point ( points, image );

  /* Step 3: Find more web lines. */
  find_more_line ( morepoints, linendx, points, image );

  /* Step 4: Change image coordinate system to Cartesian coordinate
            system.                                               */
  change_coord ( points, morepoints, linendx );

  /* Step 5: Calculate the center of gravity. */
  center_of_gravity ( centerpoints, points, morepoints, linendx );

  /* Step 6: Outputs data points to file argv[2]. */
```

```
	fileout = write_image ( argv[2], points, morepoints, linendx,
				centerpoints );

	fclose ( fileout );
  free ( (void **)image );
}

/*************************** read_image() ***************************/
/* Synopsis - Reads the intensity of every point from the image file
   and stores it to a two dimensional array. It returns the address
   of the array.

 * Parameters:
   filename - name of the input image file

 * Variables:
   1) fp - file pointer that points to the input image file
   2) data - address of a two dimensional array that stores intensity
*/
unsigned char **read_image ( char *filename )
{
	FILE *fp;
	unsigned char **data;
	int  i, j;

	if ( (fp = fopen ( filename, "rb" )) == NULL )
	{
		printf ( "Can't open file %s...\n", filename );
		exit ( 2 );
	}

  /* Allocate memory */
	if ( (data =
	(unsigned char **)calloc(ROWS, sizeof(unsigned char *)))== NULL )
		{
		printf ( "Unable to get space for pointers...\n" );
		exit ( 3 );
		}

	/* Allocate memory to store 480*640 points from the image file.
	Read intensity and store them.
	*/
	for ( i = 0; i < ROWS; i++ )
	{
		if ( (data[i]=(unsigned char *)calloc(COLUMNS,1)) == NULL )
		{
		printf ( "Unable to get space for instensity...\n" );
		exit ( 3 );
		}

		for ( j = 0; j < COLUMNS; j++ )
			if ( fread ( &data[i][j], 1, 1, fp ) != 1 )
			{
				printf ( "Error in reading input file...\n" );
				exit ( 4 );
			}
	}
```

```
	fclose ( fp );
	return ( data );
}

/************************** find_point() ******************************/
/* Synopsis - Finds out the points whose intensity is the largest. The
              method is to search both upward and downward.

 * Parameters:
   1) point - array to store the intensity of points that are blackst
   2) data - address of the two dimensional array that stores the
             intensity of every point in image file

 * Variables:
   1) mid - position y of the first point located about in the middle
of the curve
   2) row - initial position x of the first point, set by looking at
the image picture
   3) firstnum - scope of search for the first point
   4) num - scope of search for the rest points
*/
void find_point ( pointtype point[], unsigned char **data )
{
	int mid = COLUMNS/2;
	int row;
  int firstnum, num;
  int i, j;

  printf ( "Please input the position of the first point (column=320):
" );
  printf ( "row=" );
  scanf ( "%d", &row );

  printf ( "Please input the scope of search for the first point " );
  printf ( "(number of pixels): " );
  scanf ( "%d", &firstnum );
  printf ( "Please input the scope of search for the rest points " );
  printf ( "(number of pixels): " );
  scanf ( "%d", &num );

	/* Element y of each point is equal to its index in the array */
	for ( i = 0; i < COLUMNS; i++ )
		point[i].y = i;

	/* Initialize the first point */
  point[mid].x = row;
	point[mid].ints = data[row][mid];

	/* Find the first point */
	for ( i = 1; i <= firstnum; i++ )
	{
		/* Search upward */
		if ( data[row-i][mid] < point[mid].ints )
		{
			point[mid].ints = data[row-i][mid];
			point[mid].x = row-i;
```

```
        }

        /* Search downward */
        if ( data[row+i][mid] < point[mid].ints )
        {
                point[mid].ints = data[row+i][mid];
                point[mid].x = row+i;
        }
}

/* Search to the left of the first point. */
for ( j = 1; (mid-j) >= 0; j++ )
{
        /* Initialize the point */
        point[mid-j].x = point[mid-j+1].x;
        point[mid-j].ints = data[point[mid-j].x][mid-j];

        for ( i = 1; i <= num; i++ )
        {
/* Search upward */
                if ( data[point[mid-j+1].x-i][mid-j] < point[mid-
                j].ints )
                {
                        point[mid-j].ints = data[point[mid-j+1].x-
                        i][mid-j];
                        point[mid-j].x = point[mid-j+1].x-i;
                }

/* Search downward */
                if ( data[point[mid-j+1].x+i][mid-j] < point[mid-
                j].ints )
                {
                        point[mid-j].ints = data[point[mid-
                        j+1].x+i][mid-j];
                        point[mid-j].x = point[mid-j+1].x+i;
                }
        }
}

/* Search to the right of the first point. */
for ( j = 1; (mid+j) < COLUMNS; j++ )
{
        /* Initialize the point */
        point[mid+j].x = point[mid+j-1].x;
        point[mid+j].ints = data[point[mid+j].x][mid+j];

        for ( i = 1; i <= num; i++ )
        {
/* Search upward */
                if ( data[point[mid+j-1].x-i][mid+j] <
                point[mid+j].ints )
                {
                        point[mid+j].ints = data[point[mid+j-1].x-
                        i][mid+j];
                        point[mid+j].x = point[mid+j-1].x-i;
                }
```

```
                    /* Search downward */
        if ( data[point[mid+j-1].x+i][mid+j] < point[mid+j].ints )
                    {
                            point[mid+j].ints = data[point[mid+j-
                            1].x+i][mid+j];
                            point[mid+j].x = point[mid+j-1].x+i;
                    }
                }
        }
}

/*********************** find_more_line() **************************/
/* Synopsis - Finds out points on curve close and parallel to the
blackst curve.

 * Parameters:
   1) morepoint - two dimensional array to store black points of other
                  web lines
   2) linenum - scope of web lines
   3) point - array containing black points of the central/edge web
line
   4) data - address of the two dimensional array that contains image
      intensity
 */
void find_more_line ( pointtype morepoint[][COLUMNS], int linenum,
                      pointtype point[], unsigned char **data )
{
  int i, j;

  for ( i = 1; i <= linenum; i++ )
    for ( j = 0; j < COLUMNS; j++ )
    {
      /* downward lines */
      morepoint[2*i-2][j].x = point[j].x + i;
      morepoint[2*i-2][j].y = j;
      morepoint[2*i-2][j].ints = data[morepoint[2*i-2][j].x][j];

      /* upward lines */
      morepoint[2*i-1][j].x = point[j].x - i;
      morepoint[2*i-1][j].y = j;
      morepoint[2*i-1][j].ints = data[morepoint[2*i-1][j].x][j];
    }
}

/************************ change_coord() ***************************/
/* Synopsis - Changes the coordination system to represent points in
the normal way so that a curve may be drawn in the future.

 * Parameters:
   1) point - array that contains the points whose intensity is the
   largest
   2) morepoint - two dimensional array that stores black points of
   other web lines
   3) linenum - scope of web lines
*/
void change_coord ( pointtype point[], pointtype morepoint[][COLUMNS],
                    int linenum )
```

```
{
      int i, j;

  for ( j = 0; j < COLUMNS; j++ )
  {
    point[j].y = ROWS-point[j].x;
    point[j].x = j;

    for ( i = 0; i < (2*linenum); i++ )
    {
      morepoint[i][j].y = ROWS-morepoint[i][j].x;
      morepoint[i][j].x = j;
    }
  }
}

/*********************** center_of_gravity() ************************/
/* Synopsis - Calculates the center of gravity.

 * Parameters:
   1) centerpoint - array to store the center of gravity
   2) point - array that stores the central/edge web line
   3) morepoint - array that stores other web lines
   4) linenum - scope of web lines

 * Variables:
   1) divisor - divisor in the formula to calculate center of gravity
   2) dividend - dividend in the formula to calculate center of gravity
*/
void center_of_gravity ( struct center centerpoint[],pointtype point[],
                         pointtype morepoint[][COLUMNS], int linenum )
{
  int divisor, dividend;
  int i, j;

  for ( j = 0; j < COLUMNS; j++ )
  {
    dividend = 255 - point[j].ints;
    divisor = point[j].y * dividend;

    for ( i = 0; i < (2*linenum); i++ )
    {
      dividend += 255 - morepoint[i][j].ints;
      divisor += morepoint[i][j].y * (255 - morepoint[i][j].ints);
    }

    centerpoint[j].y = (float)divisor/(float)dividend;
    centerpoint[j].x = j;
  }
}

/************************** write_image() ***************************/
/* Synopsis - Stores the final points to a file.

 * Parameters:
   1) filename - name of the output file
```

```
   2) point - array containing black points of the central/edge web
      line
   3) morepoint - two dimensional array containing black points of
      other web lines
   4) linenum - scope of web lines
   5) centerpoint - array that stores the center of gravity

 * Variables
   fp - file to store data output
*/
FILE *write_image ( char *filename, pointtype point[],
                    pointtype morepoint[][COLUMNS], int linenum,
                    struct center centerpoint[] )
{
      FILE *fp;
      int i, j;

      if ( (fp = fopen ( filename, "w" )) == NULL )
      {
            printf ( "Can't open file %s...\n", filename );
            exit ( 2 );
      }

      for ( j = 0; j < COLUMNS; j++ )
  {
    fprintf ( fp, "%5d%5d%5d", point[j].x, point[j].y, point[j].ints );

    for ( i = 0; i < (2*linenum); i++ )
     fprintf ( fp, "%5d%5d", morepoint[i][j].y, morepoint[i][j].ints );

     fprintf ( fp, "%9.4f\n", centerpoint[j].y );
  }

  return ( fp );
}
```

Source SAS Code for Curve Fitting

```
dm 'log; clear; output; clear; ';
options ls=72;

data one;

* 12.dat is the .txt format file which include the deflection Z ver. X  in inch
 infile 'c:\temp\12.dat';
input x y;

x1=x;
x2=x*x;
x3=x2*x1;
x4=x2*x2;
x5=x3*x2;
x6=x3*x3;
x7=x4*x3;

x8=x4*x4;
x9=x5*x4;
x10=x5*x5;
x11=x6*x5;
x12=x6*x6;
x13=x7*x6;

y1 = x-5;
y2 = y1*y1;
y4 = y2*y2;
y6 = y4*y2;
y8 = y4*y4;
*proc print;

proc reg;
* The REG procedure will allow you to run more than one model *;
* statement *;
* The R option on the MODEL statement will compute the residuals and*;
* print them*;

*model y=x1 x2 x3 x4 x5 x6 x7 x8  ;
model y = y2 y4 y6 y8/ r;

* An output data set named TWO is created in the next step.  *;
* It contains all *;
```

```
* of the information in ONE plus the residuals and the predicted *;
* values for the *;
* MODEL statement that runs immediately before this step *;

OUTPUT OUT=two PRED=p R=r;

PROC SORT DATA=two; BY x;
* The GPLOT procedure is a high resolution graphic procedure*;
* The following GPLOT generates the predicted curve and a plot of the *;
* observed data on a single graph *;
PROC GPLOT DATA=two;
PLOT (p y)*x / OVERLAY;
symbol1 value=none l=1 cv=black i=join;
symbol2 value=point i=none cv=red ;


* The following GPLOT generates two residual plots. *;
* If the model fits the data well, the residuals will fall in a random *;
* scatter *;
* about the reference line at zero *;
PROC GPLOT DATA=two;
PLOT r*(x p) / VREF=0;
SYMBOL VALUE=POINT I=NONE;

* High resolution graphics do not go to the output window but to a * ;
* graphics window *;
* To print these graphs, do NOT use the print button at the top of *
* the screen. Use *;
* FILE - PRINT pull down menus and select your printer *;
run;
```

2

Vita

Xiangming Fang

Candidate for the Degree of

Master of Science

Thesis: MEASUREMENTS OF WEB DEFLECTION FOR THE DETERMINATION OF STABILITY OF A WEB

Major field: Mechanical Engineering

Biographical:

Personal Data: Born in China, On August 31, 1969, the son of Zaikun Fang and Xiwei Zhang

Education: BSME, East China University of Chemical Technology, China, July 1991. Completed the requirements for the Master of Science degree at Oklahoma State University in May, 1999 graduation.

Experience: Employed as Technical Manager in Deutz AG (Shanghai Office) in Shanghai, China from February 1996 to August 1997. Employed as Mechanical Engineer in Hangzhou No.1 Freezer Plant in China from August 1991 to August 1993

Professional Membership: ASME student member