1-19-94
E-8203

NASA Technical Memorandum 105271

Holographic Testing of Composite Propfans for a Cruise Missile Wind Tunnel Model

Christopher J. Miller
Lewis Research Center
Cleveland, Ohio

January 1994

NASA

HOLOGRAPHIC TESTING OF COMPOSITE PROPFANS FOR A CRUISE MISSILE WIND TUNNEL MODEL

Christopher J. Miller
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135

SUMMARY

Each of the approximately 90 composite propfan blades constructed for a 55 percent scale cruise missile wind tunnel model were holographically tested to obtain natural frequencies and mode shapes. These data were used not only for quality assurance, but also to select sets of similar blades for each blade row. Presented along with the natural frequency data is a description of a computer-based image processing system developed to supplement the photographic-based system for holographic image analysis and storage. The new system is quicker and cheaper, the holograms are indexed better, and several engineers can access the data simultaneously. The only negative effect is a slight reduction in image resolution, which does not influence the end use.

BACKGROUND

NASA Lewis Research Center uses holographic bench testing of turbomachinery components to obtain the components' natural mode shapes and frequencies. On the bench, the parts are excited via an acoustic driver which controls amplitude and a range of frequencies. These data are used to verify the dynamic characteristics and quality of fabricated components. These natural mode data are also used in the design process so that the final design part has the maximum safe operating envelope (considering integer order crossings and flutter boundaries).

The propfan blades, designed and constructed for a 55 percent scale wind tunnel model of a cruise missile, were constructed of composites instead of metal to meet wind tunnel safety requirements. Figure 1 shows a view of the lower tip speed blades, while figure 2 shows the higher tip speed blades. The blades used in this test were the first major composite fabrication project completed in-house. As a result, there were concerns about the repeatability and quality of the fabrication process as well as the accuracy of the structural modeling of the all composite parts. This report will present the holography data: natural frequencies for the first two bending and first two torsion modes, summaries of the frequencies presented as nondimensional differences from the mean values, and details of the computerized testing process. The analytic predictions corresponding to these data are presented in references 1 and 2. Details of the construction of the composite blades are documented in references 3 and 4.

Because so many holograms were desired (90 blades with at least 9 modes per blade for a total of over 810 images) a new holographic image process was developed to augment the existing photographically-based system. The new system, which consists of a video camera, digital frame grabber, and a computer for image processing, is not only faster and cheaper but also improves the engineers' ability to use the data. An overview of this system is shown in figure 3, with details discussed in appendix A.

MEASURED FREQUENCIES

The blade frequencies of interest for integer order crossing comparisons are the first two bending and first two torsional modes, denoted as 1B, 2B, 1T, and 2T respectively. These frequencies are summarized in tables I and II for CM1D (Cruise Missile blade, design 1, version D) forward and aft blades, and tables III and IV for CM2D.

For most of the blades, the measured frequencies are very consistent. The standard deviation for each natural frequency was calculated and is a small fraction of the mean, usually about 1 percent. For use in the wind tunnel experiment, sets of blades with similar frequency characteristics are desirable. To help in selecting these sets, plots of blade frequency by mode were generated along with a summary plot with normalized frequencies for all four modes of interest. To put all modes on a single summary plot, the frequencies for each mode were normalized using the mean and standard deviation

$$\tilde{f} = (f - \bar{f})/\sigma$$

where

$\tilde{f}$ = normalized natural frequency

f = measured natural frequency

$\bar{f}$ = mean natural frequency

σ = standard deviation of the natural frequency

One of the criteria for selecting blade sets was consistency of the frequencies; a "good" blade has all frequencies clustered together on the summary plot, while a "bad" blade would have one or more frequencies significantly different from the others. When a blade had frequencies that fell more than two standard deviations away from the mean, it was considered flawed and not used as part of a flight set. A blade set consists of six blades for initial installation, with two similar blades as spares.

Due to integer order crossings and flutter considerations for these blades, the significance of an out-lying mode was prioritized as 1B, 1T, 2B, and 2T. Once the blades were roughly grouped, the holograms were checked to verify mode shape similarity for the first four modes. Finally, the blade sets were grouped by frequencies and the highest frequency blade sets were used first to give the most integer order clearance. Within each blade set (including spares) the variation of natural frequencies was less than 1 percent.

During wind tunnel testing, two blades displayed obvious frequency changes from fatigue damage, a "ping" of the blade with a coin or finger produced a ringing sound that was qualitatively deader than that produced by pinging the other blades. These blades were removed from the model and retested to measure the change in frequencies and mode shapes. The fatigued blade frequencies are also listed in tables I and II.

Natural frequency plots are presented in figures 4 to 23. The individual mode frequency plots (1B, 1T, 2B, 2T) are shown first for each blade design: CM1D forward in figures 4 to 7, CM1D aft in figures 8 to 11, CM2D forward in figures 12 to 15, and CM2D aft in figures 16 to 19. The summary plots showing all frequencies for modes of all blades of each design are in figures 20 to 23 for CM1DF, CM1DA, CM2DF, and CM2DA respectively.

Representative mode shapes (1B, 1T, 2B, 2T) for each for the four blade designs are shown in figures 24 to 27. In these composite images, the images are arranged by increasing mode frequency with the lowest frequency mode in the bottom left, the next in the bottom right, the next in the top left, and the highest frequency mode in the top right. For the CM1D blades the order of increasing mode frequency is (1B, 2B, 1T, 2T) while for the CM2D blades the order is (1B, 1T, 2B, 2T). The mode shapes of the fatigued blades change slightly and are shown in figures 28 (CM1DF022) and 29 (CM1DA008). The higher frequency modes show the greatest change.

CONCLUDING REMARKS

Each of the composite propfan blades designed and constructed for a model scale cruise missile wind tunnel test program was holographically tested. The natural frequencies and mode shapes for the first eight blade modes were determined. These data were used to correlate with the structural analyses and verify the quality of the blades, as well as to select blade sets for the wind tunnel model.

During this work a computer-based image processing system was implemented to supplement a photographically-based system. The new system has these advantages

(1) Holographic images are delivered to the requester within hours after they have been taken.

(2) The holograms have the relevant test conditions stamped on the image to simplify the correlation with the test log.

(3) Several engineers can access the images and data simultaneously.

(4) The problems with filing/refiling time and space for the photos have been eliminated.

(5) It is cheaper, not only in creating the image but in any subsequent hard copy.

The only negative effect is a slight reduction in image resolution, which does not influence the end use.

ACKNOWLEDGMENT

The author would like to thank Mr. Kenneth Weiland for his help in teaching him the mechanics of creating a hologram and patience during the development of the software for this project.

APPENDIX A—COMPUTER BASED HOLOGRAPHIC IMAGE PROCESSING

OVERVIEW

Photographic holograms, the traditional method, are time consuming but are very high quality (fig. 30). When image resolution is not a high priority, a new computer-based system can provide holograms quickly and at a lower cost. The computer-based system has the additional advantages of requiring much less storage space, improving the indexing and retrieval process, and enabling several engineers to access the data at the same time.

BACKGROUND

The traditional technique for obtaining the holograms of mode shapes is a multiple step process.

<u>Step</u>	<u>Time</u>	<u>Description</u>
(1)	5 min	Paint the model with titanium dioxide paint for good reflectivity.
(2)	2 min	Mount the blade on the bench (fig. 3).
(3)	3 min	Expose and develop the reference hologram. The reference hologram, placed in front of the camera, will create an interference fringe pattern with the laser light reflected from the vibrating blade.

Steps (4) and (5) are repeated for each vibration mode of interest.

(4)	1 min	Search for a vibration mode with the acoustic driver. This involves scanning the frequency until a mode is observed, through the TV camera in the film holder, by changes in the fringe pattern.
(5)	3 min	With the part vibrating, expose and develop the time average hologram.

To create a print of the hologram for the requesting scientist to analyze the mode shape then requires

(6)	5 min	Expose a photographic negative.
(7)	? days	Send the photographic negatives to the photo lab for developing and making 8- by 10-in. prints.

The need for photographic negatives, the delay in processing through the photo lab, the filing and indexing difficulties, and the large number of holograms to create indicated a need for a new technique.

COMPUTER-BASED IMAGE PROCESSING

Since the holography system uses a video camera for the detection of the vibration modes, incorporating a PC-based frame grabber was a simple step. The video camera shown in figure 3 remains connected to the monitor for mode searching, but it is also connected to the frame grabber in the PC. The frame grabber currently being used has a resolution of 480 scan lines by 512 pixels per scan line. Each pixel can be scanned as eight bits in each of the red, green, and blue channels. Since this is a black and white camera, only one channel is scanned. A scanned image is therefore 512 by 480 pixels with 256 possible gray levels available per pixel; uncompressed, the data for the image itself totals 245,760 bytes.

The photographic process has a much finer spacial resolution than the frame grabber's 512 by 480, but this level of detail is not needed here. Similarly, the 256 gray levels are more than adequate for mode shape verification.

The computer allows for additional functionality over the photographic system since each image is stamped with a descriptive name and the relevant test data before it is saved. The images are thereby much easier to track and correlate with the test log.

Once a series of images has been obtained and stored on the PC, the images can be studied on the PC, or transferred over the local ethernet network to a workstation for the added processing power and storage available there. The workstation is also the access point for the archival storage system for computer data.

IMAGE PROCESSING STEPS

The processing of the images involves basically three steps.

(1) Determine the sub-range, within 0 to 255, that contains the useful data.
(2) Create the single mode images.
(3) Create a combined image of all modes for a given blade.

Custom software was developed to operate the frame grabber board, see appendix B and reference 5. This software currently has no contrast or brightness controls. Instead, a set of default settings are used that are conservative enough to prevent pixel value clipping. Due to the variability of the holographic imaging process, these default settings produce images with a fair amount of background noise and a dynamic range that is less than full scale. Besides automatic scaling, multiple frames could be averaged to create a time average hologram. The averaging improves the pixel signal/noise ratio.

The first processing step identifies the portion of the data range that produces the subjectively best image. This can be done iteratively for each image but a good algorithm is to select for the low end the pixel value such that 25 percent of the total pixel population lies below, and for the upper end such that 95 percent of the pixels lie below. In the PC program that calculates the histogram for an image (appendix B), this scheme is used to recommend processing limits (fig. 31). In the PC program that displays the images (appendix C), this scheme is used for the default scaling prior to display.

For each blade, ten images were created: the reference image, eight mode images, and one composite image. The sub-range expansion and single mode image creation can be done on the PC, but were usually performed on a workstation to off-load the PC. Figure 32 shows a raw image without the sub-range image expansion. Figure 33 shows the processed image enhanced with sub-range expansion.

The composite image is composed of the reference image and the eight mode images in a 3 by 3 array. The individual images are reduced from their original 512 by 480 size to 256 by 240 prior to the assembly step. The composite image size is then 768 by 720 pixels. Figure 34 shows a representative composite image. Once images have been created, they can be studied on the computer or, if a hardcopy is needed, they can be printed on a laser printer. This hardcopy method is much cheaper and easier than making the traditional 8- by 10-in. photographic prints.

IMAGE FILE COMPRESSION

The images were originally stored on the workstation in a "run-length encoded" format and then compressed with the UNIX (tm) "compress" utility to further reduce their size. The total storage needed for the images of all blades is approximately 140 MB. The size of the images (as well as the total storage needed) is a major problem for the PC-based image processing scheme. During the holography, the images rapidly fill the hard disk on the PC and require substantial time to off-load.

An alternative storage scheme considered is GIF (Graphic Interchange Format, (c) CompuServe), which includes an internal Lempel-Zif-Welch compression scheme and can store images with up to 256 unique colors (or 256 levels of grey-scale). As noted in table VII, both schemes achieve about the same compression. For grey-scaled images stored in a format that preserves all the pixel information, this is about as compressed as possible. Since the actual pixel values in the image are qualitative rather than quantitative, a compression scheme that loses some information in exchange for greater compression can be used.

The Joint Photographic Experts Group (JPEG) scheme (ref. 6) is one such lossy technique. The heart of the JPEG scheme is the use of a Discrete Cosine Transform on patches of the image. As the quality level is increased, more of the high order, high spacial frequency coefficients are included. For images with smooth spacial variations, only a few coefficients need to be saved, resulting in a large compression ratio with a corresponding increase in the loss of high spacial frequency information. In the holographic images, high JPEG compression (low Q factors) primarily introduces errors at the edges of the labels stamped on to the images.

Table VII summarizes the compression ratios achieved with the various schemes, as well as the JPEG scheme with varying loss factors. Subjectively evaluating the images as displayed on the computer reveals that the image compressed with JPEG at $Q = 50$ looks to the eye like the original image. Even more interesting is that the original image printed in 16 grey levels (fig. 29) looks the same as one compressed with JPEG at $Q = 20$ in a 16 grey level print (fig. 30).

If JPEG compression at $Q = 50$ is used to process all the images, the total storage needed would drop from 140 MB down to approximately 8 MB, removing the storage problems of the PC computer-based technique.

APPENDIX B—LISTING OF FRAME GRABBER FORTRAN PROGRAM

```
$INCLUDE:'FORINTF.H'

        PROGRAM DRVPIP
C**********************************************************************
C
C  SIMPLE DRIVER PROGRAM FOR THE PIP BOARD
C
C  THIS IS A GENERIC PROGRAM TO TAKE HOLOGRAPHIC IMAGE DATA
C
C  910204 CHRISTOPHER J. MILLER
C
C**********************************************************************
        IMPLICIT  INTEGER (A-E)
        IMPLICIT  REAL (F)
        IMPLICIT  INTEGER (G-Z)
C     INTEGER*4 IHISTO
C     INTEGER*2 DECODE
C
      CHARACTER STRING*21, FNAME*13, DATSTR*9
C     CHARACTER TIMSTR*9
      CHARACTER IMGLBL*8, ANS*1
      INTEGER   I
      INTEGER*2 IVAL
      INTEGER   FREQ, VOLT
      CHARACTER BUFFER(4096)
C
C interpreter commands:
C inifmt 26c 1 0 0 0 0
C setind 255
C chan 2
C sync 1
C quadmode 4
      IVAL = INIFMT(620, 1, 0, 0, 0, 0)
      IF (IVAL.NE.1) THEN
        PRINT *,'PIP board not found; return value =',I
        STOP
      ENDIF
      IVAL = IWINMD(0)
C     CALL SETWIN(0, 0, 511, 511)
      CALL SETIND(255)
      CALL SYNC(1)
      CALL CHAN(2)
      CALL DELAY(5)
C
C Setup: select a good gain and offset
      CALL AUTO
      CALL CLEAR(0,7)
      CALL SNAP(1)
      PRINT *,'Do you want to set the gain and offset?'
      READ (5,'(A)') ANS
      IF (ANS.EQ.'Y' .OR. ANS.EQ.'y') CALL SETUP
C
C LOOP FOR TAKING DATA
  100 CONTINUE
        PRINT *
        WRITE(6,1000)
 1000   FORMAT(' ¦......¦ UNIQUE LABEL (FILE NAME) FOR MODE IMAGES')
        READ (5,'(1A8)') IMGLBL
        CALL UPCASE(IMGLBL)
        PRINT *
C MODE LOOP
  105   CONTINUE
        WRITE(6,1010)
 1010   FORMAT($,' MODE (-1 TO QUIT) : ')
        READ (5,*) MODE
        IF (MODE.LT.0) GOTO 150
C
          PRINT *,'PRESS ENTER WHEN THE IMAGE IS READY'
          READ (5,*)
```

```
C
C interpreter commands:
C clear 0 7
C snap 1
C sh1 0 0
C moveto 50 400
C text 2 "CM-1D F #14 779HZ 9.88V"
C todisk 4096 0 sample.fil 5200 -1
C
          CALL CLEAR(0,7)
          CALL SNAP(1)
C
C GET THE FREQUENCY AND VOLTAGE
          WRITE(6,1020)
 1020     FORMAT($,'    ENTER THE FREQUENCY: ')
          READ (5,*) FREQ
          WRITE(6,1030)
 1030     FORMAT($,'    ENTER THE VOLTAGE  : ')
          READ (5,*) VOLT
C
C GENERATE THE FILE NAME
C LABEL THE IMAGE WITH THE (FILE NAME) + MODE & EXCITATION VOLTAGE
          WRITE(FNAME,1040) IMGLBL,MODE
 1040     FORMAT(A8,'.',I3)
          DO 110 I=1,8
            IF (FNAME(I:I).EQ.' ') FNAME(I:I)='_'
  110     CONTINUE
          DO 120 I=10,12
            IF (FNAME(I:I).EQ.' ') FNAME(I:I)='0'
  120     CONTINUE
          FNAME(13:13) = CHAR(0)
          WRITE(STRING,1050) FNAME,VOLT
 1050     FORMAT(A12,I3,'V')
          STRING(17:17) = CHAR(0)
C
C WRITE TEXT IN WHITE (255)
          CALL SETIND(255)
          CALL MOVETO(50, 410)
          CALL TEXT(STRING,3)
C
C LABEL THE IMAGE WITH THE DATE AND FREQUENCY
          CALL DATE(DATSTR)
C         CALL TIME(TIMSTR)
C         WRITE(STRING,1060) DATSTR,TIMSTR
          WRITE(STRING,1060) DATSTR, FREQ
 1060     FORMAT(A8,' ',I5,'Hz')
          DO 125 I=1,8
            IF (STRING(I:I).EQ.' ') STRING(I:I)='0'
  125     CONTINUE
          STRING(17:17) = CHAR(0)
          CALL MOVETO(50, 440)
          CALL TEXT(STRING,3)
C
C SAVE THE IMAGE IN THE FILE
          IVAL = IWINTO(4096, FNAME, BUFFER)
          IF (IVAL.EQ.0) THEN
            PRINT *,'IWINTO ERROR CODE 0'
            PRINT *,'        COULD NOT OPEN FILE'
            PRINT *,'        FILE NAME IS:',FNAME
            STOP
          END IF
          IF (IVAL.EQ. -1) THEN
            PRINT *,'IWINTO ERROR CODE -1'
            PRINT *,'        TRANSFER TERMINATED PREMATURELY'
            PRINT *,'        (PROBABLY A FULL DISK)'
            PRINT *,'        FILE NAME IS:',FNAME
            PRINT *
            PRINT *,'        CHECK THE DISK AND TRY AGAIN'
            STOP
          END IF
          GOTO 105
C END OF MODE LOOP
```

```
  150   CONTINUE
        PRINT *
        WRITE(6,1070)
 1070   FORMAT($,' ANOTHER TEST OBJECT? (Y/N) ')
        READ (5,'(A)') ROW
        IF (ROW.EQ.'Y' .OR. ROW.EQ.'y') GOTO 100
C
      CALL PEXIT
      STOP
      END
      SUBROUTINE DELAY(I)
C*************************************************************************
C
C  A DELAY IN TERMS OF "I" VERTICAL RETRACES
C
C*************************************************************************
      DO 100 N = 1, I
        CALL VWAIT()
  100 CONTINUE
      RETURN
      END
      SUBROUTINE SETUP
C*************************************************************************
C
C  SET UP THE GAINS, ETC. TO GET A GOOD IMAGE TO START WITH
C
C*************************************************************************
      CHARACTER ANS*1
      INTEGER   OSV, GV
C
      PRINT *,'The optimal gain and offset for A/D conversion have'
      PRINT *,'been set.  This routine allows you to change them to'
      PRINT *,'get a better image for the actual data.  Note that if'
      PRINT *,'the image is clipped here, it cannot be fixed with'
      PRINT *,'post processing.'
      PRINT *
      PRINT *,'Adjust offset first, then gain in a separate pass.'
      PRINT *

      PRINT *,'PRESS ENTER WHEN SOME IMAGE IS READY'
      READ (5,*)
      CALL CLEAR(0,7)
      CALL SNAP(1)
  100 CONTINUE
        PRINT *,'ADJUST: (Offset, Gain, Exit)'
        READ (5,'(A)') ANS

        IF (ANS.EQ.'O' .OR. ANS.EQ.'o') THEN
          PRINT *,'Enter OFFSET: 0 (darker) - 255 (lighter)'
          READ (5,*) OSV
          OSV = AMIN0(255,AMAX0(0,OSV))
          CALL OFFSET(OSV)
          CALL CLEAR(0,7)
          CALL SNAP(1)
        ENDIF

        IF (ANS.EQ.'G' .OR. ANS.EQ.'g') THEN
          PRINT *,'Enter GAIN: 0 (darker) - 255 (lighter)'
          READ (5,*) GV
          GV = AMIN0(255,AMAX0(0,GV))
          CALL GAIN(GV)
          CALL CLEAR(0,7)
          CALL SNAP(1)
        ENDIF

        IF (ANS.NE.'E' .AND. ANS.NE.'e') GOTO 100
      RETURN
      END
C*************************************************************************
      SUBROUTINE DATE(S)
        CHARACTER S * 8
        CALL GETDAT(IYR, IMON, IDAY)
```

```
       WRITE(S,1000) IMON,IDAY,IYR-1900
 1000  FORMAT(I2,'/',I2,'/',I2)
       RETURN
      END
C**********************************************************************
      SUBROUTINE TIME(S)
       CHARACTER S * 8
       CALL GETTIM(IHR, IMIN, ISEC, I100TH)
       WRITE(S,1000) IHR,IMIN,ISEC
 1000  FORMAT(I2,':',I2,':',I2)
       RETURN
      END
C**********************************************************************
      SUBROUTINE UPCASE(S)
       CHARACTER S * 8
       INTEGER   I
       DO 100 I=1,8
         IF( ICHAR(S(I:I)).GT.ICHAR('a') .AND.
     >         ICHAR(S(I:I)).LT.ICHAR('z') ) THEN
           S(I:I)=CHAR(ICHAR(S(I:I))-ICHAR('a')+ICHAR('A'))
         ENDIF
  100  CONTINUE
       RETURN
      END
```

APPENDIX C—LISTING OF PC HISTOGRAM PASCAL PROGRAM

```
{$R-,S-}
program piphist;
{ Calculate and display a histogram for an image from the PIP board.
  Chris Miller  901207

  PIP image is 512x480, and upside down.

  v910701  cjm  Now uses BlockRead to speed up input.
  v910812  cjm  Tried various color schemes
}
uses
  cjm, Crt, Dos, Drivers, {Graph} HPGL;
const
  version = '910812';
  FracLow  = 0.3;
  FracHigh = 0.95;
var
  grDriver : integer;
  grMode   : integer;
  errCode  : integer;

  iFileName : string;
  iFile     : file;

  hist      : array[0..255] of word;
  integrl   : array[0..255] of real;
  maxpixel,
  minpixel  : word;
  ImgRows   : array[1..2048] of byte; { four rows a a time }
  result    : word;

  c           : char;
  i, j        : integer;
  ilow, ihigh : integer;
  xbase, ybase : integer;
  r           : real;
  s           : string;
  col         : byte;
  TextH       : integer;

procedure Abort(Msg : string);
begin
  Writeln(Msg, ': ', GraphErrorMsg(GraphResult));
  Halt(1);
end;

begin
  if (ParamCount <> 1) or (ParamStr(1) = '?') then
  begin
    writeln('Usage: piphist image.fil');
    Halt;
  end;

  iFileName := ParamStr(1);
  if not FileExists(iFileName) then
  begin
      writeln('File ',UpStr(iFileName),' not found');
      Halt;
  end;
  Assign(iFile,iFileName);
  Reset(iFile, 1);

  { Read in the data, and tabulate for the histogram }
  for i:=0 to 255 do hist[i]:=0;

  write('Processing line ');
  col := WhereX;
  write('  0 of 480.');
  for i:=1 to 480 div 4 do begin
```

```
    if (i mod 10) = 0 then begin
      GotoXY(col,WhereY);
      write(4*i:3);
    end;
    BlockRead(ifile,ImgRows,SizeOf(ImgRows),result);
    for j:=1 to 512 * 4 do
    Inc(hist[ImgRows[j]]);
  end;

  { find the maximum & minimum pixel bin values }
  maxpixel := hist[0];
  minpixel := hist[0];
  for i:= 1 to 255 do begin
    if hist[i] > maxpixel then  maxpixel := hist[i];
    if hist[i] < minpixel then  minpixel := hist[i];
  end;

  { create the integral }
  integrl[0] := hist[0] / maxpixel;
  for i:=1 to 255 do
    integrl[i] := integrl[i-1] + hist[i]/maxpixel;

  { draw the histogram }
  grDriver := Detect;
  InitGraph(grDriver,grMode,'\tp');
  SetColor(White);
  s := 'PIPHIST: Histogram plotting for PIP board output   CJM '+version;
  { base location for the rectangle }
  xbase := 50;
  TextH := TextHeight('A');
  ybase := 2;
  OutTextXY(xbase,ybase, s);
  ybase := ybase +2*TextH;

  SetColor(DarkGray);
  Rectangle(xbase-3,ybase-2, xbase+2*255+3,ybase+200+1);
  SetColor(red);
  SetLineStyle(DottedLn,0,NormWidth);
  for i := 0 to 255 do  { draw vertical grid lines every 10 counts }
    if (i mod 10) = 0 then Line(xbase+2*i,ybase, xbase+2*i,ybase+200);

  ybase := ybase+200;
  SetColor(White);
  SetLineStyle(SolidLn,0,NormWidth);
  for i := 0 to 255 do begin
    r := hist[i];
    r := 200*r/maxpixel;
    if (i mod 20)=0 then SetColor(red)
                    else SetColor(white);
    Line(xbase+2*i,ybase-trunc(r), xbase+2*i,ybase);
  end;
  MoveTo(50,220);
  SetColor(blue);
  for i := 1 to 255 do begin
    LineTo( xbase+2*i, ybase -trunc(200*integrl[i]/integrl[255]) );
  end;

  { tag ends of scale }
  SetColor(White);
  SetLineStyle(SolidLn,0,NormWidth);
  ilow := 0;
  Line(xbase+2*ilow,ybase, xbase+2*ilow,ybase +TextH div 2);
  s    := '0';
  OutTextXY(xbase+2*ilow-TextWidth(s) div 2, ybase+TextH, s);
  ilow := 255;
  Line(xbase+2*ilow,ybase, xbase+2*ilow,ybase +TextH div 2);
  s    := '255';
  OutTextXY(xbase+2*ilow-TextWidth(s) div 2, ybase+TextH, s);

  { tag recommended processing lines }
  ilow := 0;
  while (integrl[ilow] < FracLow*integrl[255]) do Inc(ilow);
```

```
  SetColor(green);
  SetLineStyle(SolidLn,0,ThickWidth);
  r := hist[ilow];
  r := 200*r/maxpixel;
  Line(xbase+2*ilow,ybase-trunc(r), xbase+2*ilow,ybase+TextH);
  s := IntToStr(ilow);
  OutTextXY(xbase+2*ilow-TextWidth(s) div 2, ybase+2*TextH, s);
  ihigh := 254;
  while (integrl[ihigh] > FracHigh*integrl[255]) do Dec(ihigh);
  SetColor(green);
  r := hist[ihigh];
  r := 200*r/maxpixel;
  Line(xbase+2*ihigh,ybase-trunc(r), xbase+2*ihigh,ybase+TextH);
  s := IntToStr(ihigh);
  OutTextXY(xbase+2*ihigh-TextWidth(s) div 2, ybase+2*TextH, s);

  { labeling }
  SetColor(White);
  ybase := ybase +4*TextH;
  s := 'Pixel population by value for the file: ' + UpStr(iFileName);
  OutTextXY(xbase,ybase, s);
  ybase := ybase +2*TextH;
  s := 'Minimum pixel (bin) count = ' + IntToStr(minpixel);
  OutTextXY(xbase,ybase, s);
  ybase := ybase +2*TextH;
  s := 'Maximum pixel (bin) count = ' + IntToStr(maxpixel);
  OutTextXY(xbase,ybase, s);
  ybase := ybase +2*TextH;
  s := 'Suggested processing range (for image expansion) is: '
      + IntToStr(ilow) + ' - ' + IntToStr(ihigh);
  OutTextXY(xbase,ybase, s);

  while KeyPressed do c:=ReadKey;
  repeat until KeyPressed;
  CloseGraph;
End.
```

APPENDIX D—LISTING OF PC IMAGE DISPLAY PASCAL PROGRAM

```
{$R-,S-}
{$define ega}
program pipdisp;
{ process and display an image from the pip board
  Christopher J. Miller 901209

  PIP image is 512x480, and upside down.

  v910701  cjm  Now uses BlockRead to speed up the input.
  v910725  cjm  In InitGraph, tp path is now 'c:\tp'; OutputHelp procedure.
  v910812  cjm  Check for <Esc> to quit.
}
uses
  cjm, Crt, Dos, Drivers, Graph;
type
  ColorMapType = array[0..15] of word;
const
  version = '910812';
  RowSize = 512;
  ColorSeq : ColorMapType
           =( Black,DarkGray,Blue,Red,Brown,Magenta,Green,Cyan,
              LightGray,LightBlue,LightRed,LightMagenta,LightGreen,Yellow,
              LightCyan, White);
  BWSeq    : ColorMapType
           =( 0,1,4,5,8,6,2,3,9,12,7,10,11,13,14,15);
{            =( Black,DarkGray,LightGray,White,LightGray,DarkGray,Black,
               Black,DarkGray,LightGray,White,LightGray,DarkGray,Black,
               DarkGray, White);}
{           =( Black,Black,Black,Black,
              DarkGray,DarkGray,DarkGray,DarkGray,
              LightGray,LightGray,LightGray,LightGray,
              White,White,White,White);}
{           =( Black,White,Black,White,Black,White,Black,White,
              Black,White,Black,White,Black,White,Black,White);}
var
  grDriver : integer;
  grMode   : integer;
  errCode  : integer;

  iFileName : string;
  iFile     : file;
  ImgRows   : array[1..2048] of byte;  { four rows at a time }
  result    : word;
  hist      : array[0..255] of word;
  integrl   : array[0..255] of real;
  pl, ph    : byte;   { low and high pixel value for img expansion }
  ph_pl     : byte;
  vskip     : byte;
  imsg      : integer;
  ColorMap  : ColorMapType;
  UseColor  : (bw,color,user);
  MapFile   : text;
  MapName   : PathStr;

  c         : char;
  i, j, k   : integer;
  iptr      : integer;
  row       : integer;
  xoffset   : integer;
  r         : real;
  s         : string;
  w         : word;
  col       : integer;

procedure Abort(Msg : string);
begin
  Writeln(Msg, ': ', GraphErrorMsg(GraphResult));
  Halt(1);
end;
```

```
procedure CalcExpansion;
{ Calculate the image expansion pixel values from the 30%/95% rule }
const
  FracLow  = 0.3;
  FracHigh = 0.95;
var
  i : integer;
  maxpixel : word;
begin
  writeln('Calculating the auto-scaling');
  Reset(iFile, 1);

  { Read in the data, and tabulate for the histogram }
  for i:=0 to 255 do hist[i]:=0;

  write('Processing line ');
  col := WhereX;
  write('  0 of 480.');
  for i:=1 to 480 div 4 do begin
    if (i mod 10) = 0 then begin
      GotoXY(col,WhereY);
      write(4*i:3);
    end;
    BlockRead(iFile,ImgRows,SizeOf(ImgRows),result);
    for j:=1 to 512 * 4 do
      Inc(hist[ImgRows[j]]);
  end;

  maxpixel := hist[0];
  for i := 1 to 255 do
    if (hist[i] > maxpixel) then maxpixel := hist[i];

  { create the integral }
  integrl[0] := hist[0] / maxpixel;
  for i := 1 to 255 do
    integrl[i] := integrl[i-1] + hist[i]/maxpixel;

  { calc pl & ph processing values }
  pl := 0;
  while (integrl[pl] < FracLow*integrl[255]) do Inc(pl);
  ph := 254;
  while (integrl[ph] > FracHigh*integrl[255]) do Dec(ph);
  writeln;
  writeln('Low value =',pl:4);
  writeln('High value = ',ph:4);
  Pause('');
end;

Procedure OutputHelp;
Begin
  writeln('Usage: pipdisp image.fil [-p:nnn] [-P:nnn] [-s:nnn] [-b] [-v]');
  writeln('Where the low (-p) and high (-P) pixel values are used to expand');
  writeln('a subrange.  If not specified, the values for -p & -P are calculated');
  writeln('from the integral of pixel bin values: the low end is at 30% of the');
  writeln('total, the high end is at 90% of the total.');
  writeln('[-s:nnn] The skip option is the number of lines at the top of the image');
  writeln('         to skip (defaults to ',vskip,').');
  writeln('[-b]     The black and white color map is used instead of the color one.');
  writeln('[-m:filename] user defined colormap');
  writeln('[-v]     Force VGA mode graphics.');
End;


begin
{$ifdef ega}
  { Register the EGAVGA driver }
  if RegisterBGIdriver(@EGAVGADriverProc) < 0 then
    Abort('EGA/VGA');
{$endif}

  pl := 255;
```

```
ph := 0;
vskip := 80;
UseColor := color;
grDriver := Detect;

if ParamCount = 0 then begin
  OutputHelp;
  Halt;
end;

for i:=1 to ParamCount do begin
  if ParamStr(i) = '?' then begin
    OutputHelp;
    Halt;
  end;
  s := ParamStr(i);
  if (s[1] = '-') then
    case s[2] of
      'p' : Val(Copy(s,4,Length(s)-3),pl,imsg);
      'P' : Val(Copy(s,4,Length(s)-3),ph,imsg);
      's' : Val(Copy(s,4,Length(s)-3),vskip,imsg);
      'b' : UseColor := bw;
      'm' : begin
              UseColor := User;
              MapName := Copy(s,4,Length(s));
            end;
      'v' : grDriver := VGA;
    end;
end;

case UseColor of
  color : ColorMap := ColorSeq;
  bw    : ColorMap := BWSeq;
  user  : begin
            Assign(MapFile,MapName);
            Reset(MapFile);
            for i := 0 to 15 do
              read(MapFile, ColorMap[i]);
          end;
end;

iFileName := ParamStr(1);
if not FileExists(iFileName) then begin
  writeln('File ',UpStr(iFileName),' not found');
  Halt;
end;
Assign(iFile,iFileName);
Reset(iFile, 1);


if (pl = 255) and (ph = 0) then
  CalcExpansion;

if (pl > ph) then  begin
  i  := ph;
  ph := pl;
  ph := i;
end;
ph_pl := ph-pl;


{ Display the image while processing }
grDriver := Detect;
InitGraph(grDriver,grMode,'c:\tp');

{ if VGA, set gray scale levels }
if (grDriver=VGA) then
for i := 0 to MaxColors do begin
  j := Trunc(255.0*MaxColors/i);
  SetRGBPalette(i, j,j,j);
end;
```

```
  SetColor(White);
  xoffset := (GetMaxX - (RowSize-1) ) div 2;

(* ImageSize for a line of 512 pixels is: *)
(* for herc: 70 bytes = 64 + 6 *)
(* for ega: 262 bytes = 256 + 6 *)

  Reset(iFile, 1);

  { skip some of the lines at the top }
  if (vskip > 8) then
  for j := 1 to (vskip div 4) do
    BlockRead(iFile,ImgRows,SizeOf(ImgRows),result);

  BlockRead(iFile,ImgRows,SizeOf(ImgRows),result);
  j := 0;
  while (result = 2048) and (j < GetMaxY) do begin
    iptr := 0;
    for row := 0 to 3 do begin
      for i:=0 to RowSize-1 do begin
        Inc(iptr);
        k := integer(ImgRows[iptr]) -pl;
        if (k < 0) then
          w := 0
        else if (k > ph-pl) then
          w := 255
        else
          w := ((k shl 8) -1) div ph_pl;
         { Now scale 0..255 to 0..15 for color selection }
        w := w shr 4;
        PutPixel(xoffset+i,j,ColorMap[w]);
      end;
      Inc(j);
    end;
    BlockRead(iFile,ImgRows,SizeOf(ImgRows),result);
  end;

  { Add colored pixel map key }
  SetColor(white);
  Rectangle(0,301,11,44);
  for j := 0 to 255 do begin
    k := j -pl;
    if (k < 0) then w := 0
               else w := (256*k-1) div ph_pl;
    if (w > 255) then w := 255;
    w := w shr 4;
    SetColor(ColorMap[w]);
    Line(1,300-j, 10,300-j);
    if (ColorMap[w] = black) then SetColor(white);
    if (j mod 16)=0 then OutTextXY(20,300-j-2,IntToStr(j));
  end;

  while KeyPressed do c:=ReadKey;
  repeat until KeyPressed;

  CloseGraph;
End.
```

REFERENCES

1. Ernst, M.A.: Structural Analysis of Low RPM Composite Propfan Blades for the LRCSW Wind Tunnel Model. NASA TM–105266.

2. Carek, D.A.: Structural Analysis of High RPM Composite Propfan Blades for a Cruise Missile Wind Tunnel Model. NASA TM–105267.

3. Thorp, S.A.; and Downey, K.M.: Computer Aided Design and Manufacturing of Composite Propfan Blades for a Cruise Missile Wind Tunnel Model. NASA TM–105269.

4. Fite, E.B: Fabrication of Composite Propfan Blades for a Cruise Missile Wind Tunnel Model. NASA TM–105270.

5. PIP Video Digitizer Board Hardware Manual #289-MH-00 Rev 5, October 11, 1989. Matrox Electronic Systems Limited, 1055 St. Regis Blvd., Dorval, Quebec, Canada H9P2T4.

6. Wallace, G.K.: The JPEG Still Picture Compression Standard. Communications of the ACM, April 1991, vol. 34, no. 4, pp. 38–44.

TABLE I.—NATURAL FREQUENCIES FOR CM1D FORWARD BLADES

SN	Natural frequencies, Hz			
	1B	2B	1T	2T
1	904	1881	2405	2883
3	823	1824	2132	2770
4	825	1778	2092	2742
5	829	1797	2133	2724
6	823	1754	2120	2674
7	823	1771	2119	2682
8	831	1798	2107	2673
9	829	1780	2119	2648
10	839	1798	2160	2693
11	843	1815	2163	2723
12	849	1839	2189	2735
15	842	1815	2149	2667
16	838	1807	2137	2691
17	843	1825	2157	2727
18	849	1834	2176	2751
19	828	1797	2119	2696
20	831	1788	2112	2674
21	841	1825	2144	2733
22	835	1797	2144	2702
23	831	1789	2132	2682
24	831	1793	2137	2702
25	838	1807	2163	2706
$\bar{f}$	834	1802	2138	2704
σ	8	22	24	25
Pred[1]	912	2006	2292	3034
22[2]	794	1711	2044	2631
25[2]	717	1611	1943	2576

[1]Predicted frequencies.
[2]Remeasured after tunnel test.

TABLE II.—NATURAL FREQUENCIES FOR CM1D AFT BLADES

SN	Natural frequencies, Hz			
	1B	2B	1T	2T
1	914	1824	2340	2721
2	893	1771	2308	2694
3	823	1762	2125	2626
6	824	1697	2126	2564
7	833	1731	2131	2570
8	824	1692	2162	2501
9	819	1677	2077	2519
10	825	1701	2138	2534
11	828	1713	2190	2566
12	840	1737	2215	2568
13	819	1700	2119	2540
14	829	1716	2189	2562
15	822	1716	2143	2600
16	834	1736	2203	2589
17	822	1698	2157	2514
18	817	1676	2112	2533
19	829	1708	2157	2543
20	817	1669	2107	2506
21	816	1709	2150	2556
22	829	1696	2142	2553
23	827	1702	2157	2555
24	821	1701	2175	2542
25	825	1712	2182	2563
$\bar{f}$	825	1707	2150	2752
σ	6	22	34	30
Pred[1]	892	1928	2300	2981
8[2]	790	1568	2011	2391
25[2]	747	1522	1923	2405

[1]Predicted frequencies.
[2]Remeasured after tunnel test.

TABLE III.—NATURAL FREQUENCIES FOR CM2D FORWARD BLADES

Natural frequencies, Hz				
SN	1B	1T	2B	2T
5	966	1689	2428	3234
6	947	1663	2398	3211
7	947	1697	2412	3220
8	950	1684	2409	3209
9	968	1697	2431	3242
10	953	1673	2407	3232
11	956	1677	2416	3226
12	958	1668	2406	3223
13	958	1669	2419	3240
14	960	1689	2434	3245
15	951	1669	2398	3213
16	938	1669	2349	3194
17	955	1683	2409	3219
18	943	1677	2382	3199
19	955	1670	2398	3193
20	963	1695	2429	3240
21	960	1680	2420	3237
22	974	1707	2452	3246
23	946	1662	2388	3182
24	940	1661	2384	3190
25	961	1689	2429	3232
26	954	1666	2399	3206
30[1]	938	1697	2405	3222
$\bar{f}$	954	1680	2409	3220
σ	10	13	22	19
Pred[2]	980	1971	2656	3615

[1]Blade used in fatigue testing.
[2]Predicted frequencies.

TABLE IV.—NATURAL FREQUENCIES FOR CM2D AFT BLADES

Natural frequencies, Hz				
SN	1B	1T	2B	2T
3	968	1777	2348	2982
5	958	1762	2330	2979
6	958	1762	2332	3000
7	956	1764	2340	2970
8	953	1768	2319	2959
9	958	1771	2318	2972
10	961	1771	2319	2963
11	948	1766	2297	2938
12	958	1777	2261	2965
13	956	1771	2329	2971
14[1]	968	1798	2464	3029
15	951	1765	2259	2941
16	952	1755	2313	2969
17	953	1754	2327	2964
18	963	1762	2364	2953
19	951	1763	2238	2940
20	954	1769	2337	2990
22	963	1771	2359	2976
23[2]	960	1767	2345	2971
24	937	1725	2293	2918
25	961	1754	2337	2947
26	953	1717	2301	2899
27	944	1746	2302	2949
28	943	1721	2292	2904
29	939	1717	2286	2877
30	946	1746	2311	2918
$\bar{f}$	954	1758	2320	2955
σ	8	20	42	33
Pred[3]	962	2038	2592	3316

[1]Image marked as #114.
[2]Image marked as #214.
[3]Predicted frequencies.

TABLE V.—CM1D BLADE SETS

Forward sets			Aft sets		
#1	#2	#3	#1	#2	#3
11	4	3	7	6	3
12	5	6	11	8	20
15	7	10	12	9	22
16	8		14	10	
17	9		15	13	
18	19		16	17	
21	20		19	18	
22	23		23	21	
25	24		25	24	

TABLE VI.—CM2D BLADE SETS

Forward sets			Aft sets		
#1	#2	#3	#1	#2	#3
6	7	5	24	8	5
15	8	9	26	11	6
16	10	14	27	16	7
18	11	20	28	17	13
19	12	21	29	20	14
23	13	22	30	25	22
24	17	25			23
26					9
					10
					18

TABLE VII.—REPRESENTATIVE COMPRESSION RESULTS

Scheme	Size	Ratio
Original image	247692	1.0:1
Compressed	175636	1.4:1
GIF	174292	1.4:1
JPEG, Q = 100	102461	2.4:1
JPEG, Q = 75	14753	16.8:1
JPEG, Q = 50	9436	26.2:1
JPEG, Q =25	5994	41.3:1
JPEG, Q = 20	5311	46.6:1
JPEG, Q = 10	3697	67.0:1

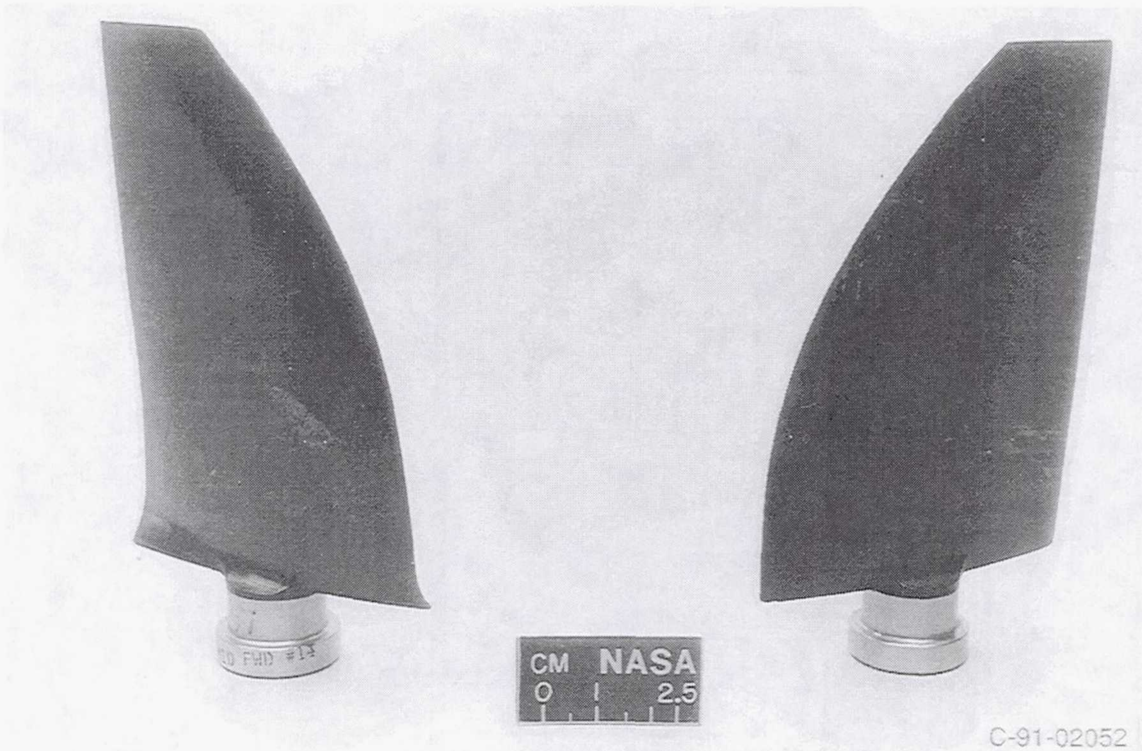

Figure 1 – CM1D: Low Tip Speed Propfan Fore and Aft Designs

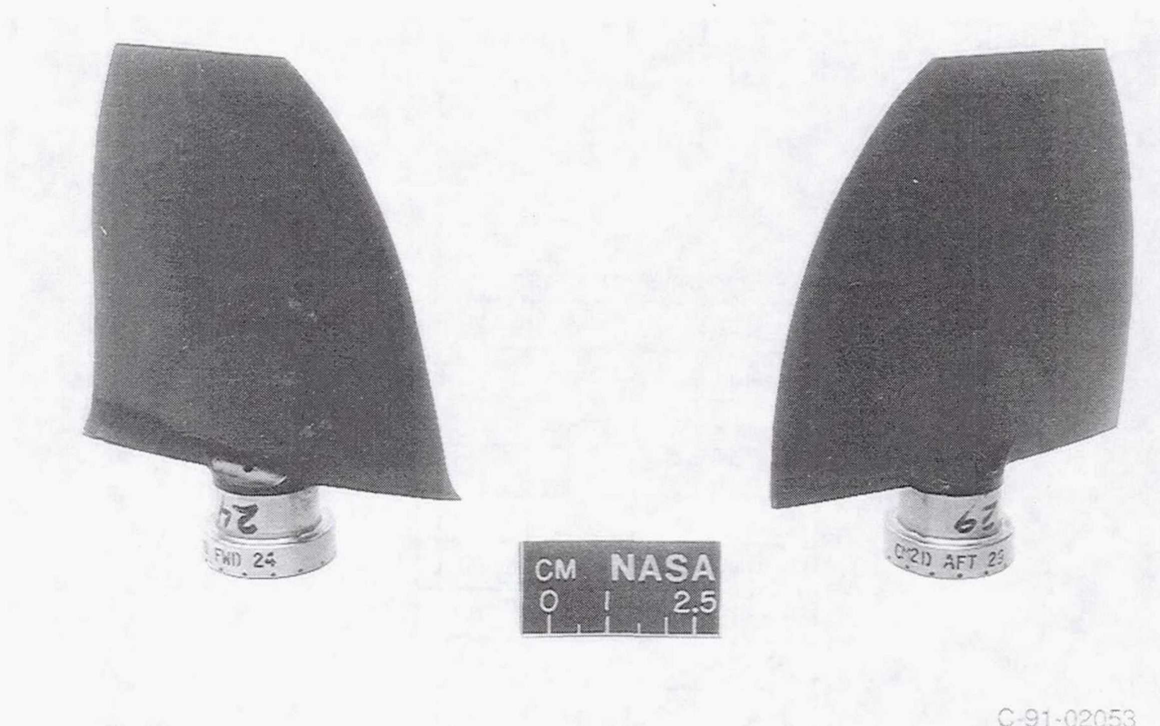

Figure 2 – CM2D: High Tip Speed Propfan Fore and Aft Designs

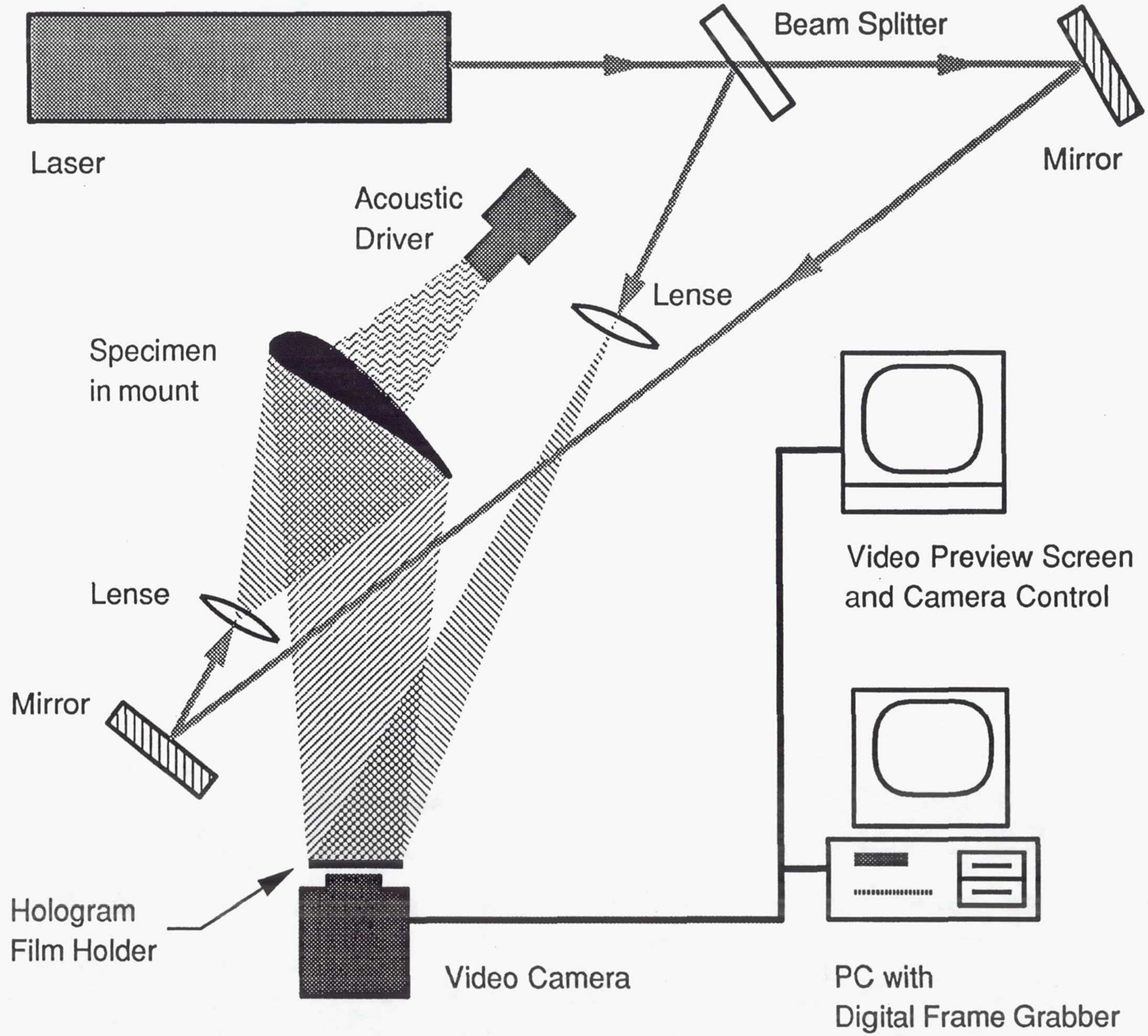

Figure 3 – Holography Setup for Computer Based Image Capture

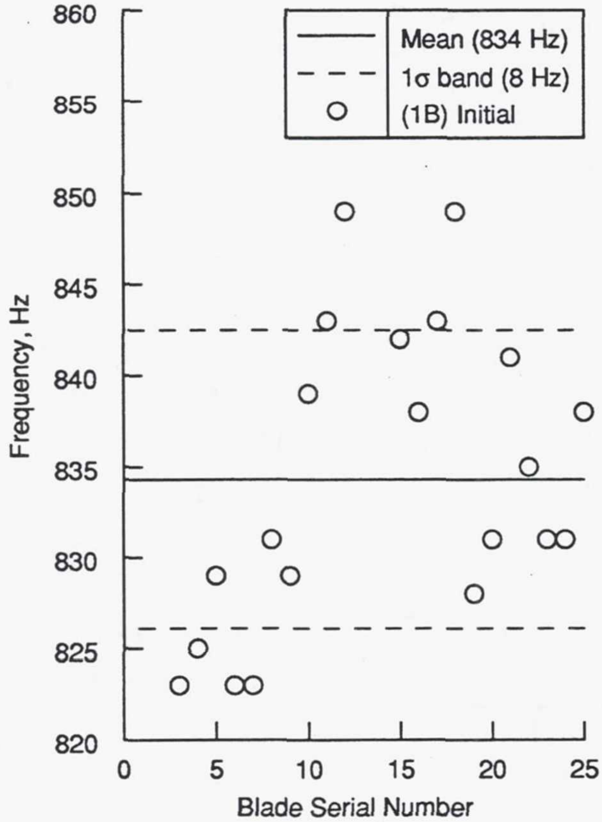

Figure 4 – CM1DF first bending mode

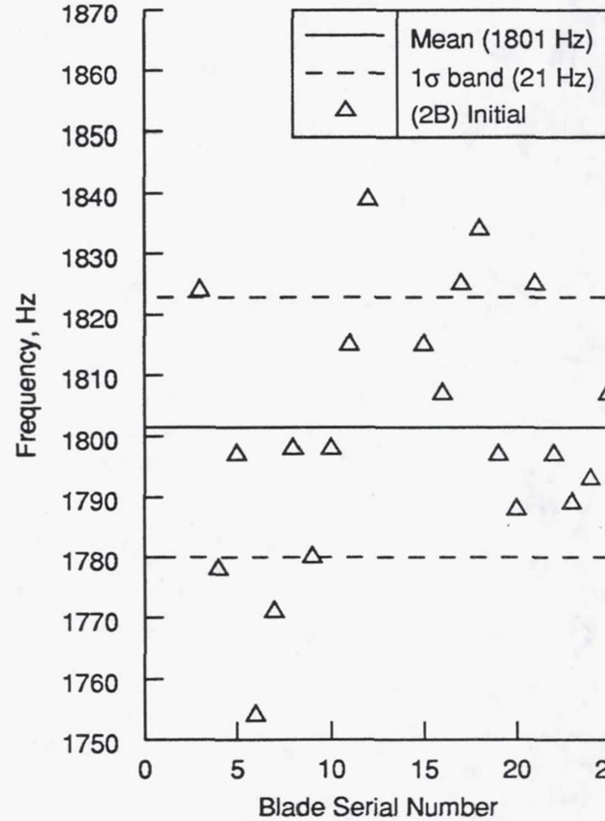

Figure 5 – CM1DF second bending mode

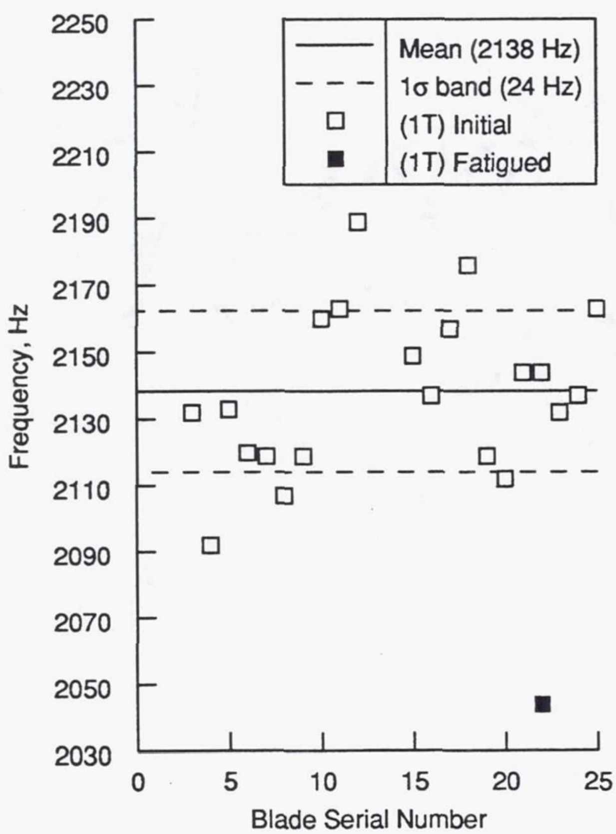

Figure 6 – CM1DF first torsion mode

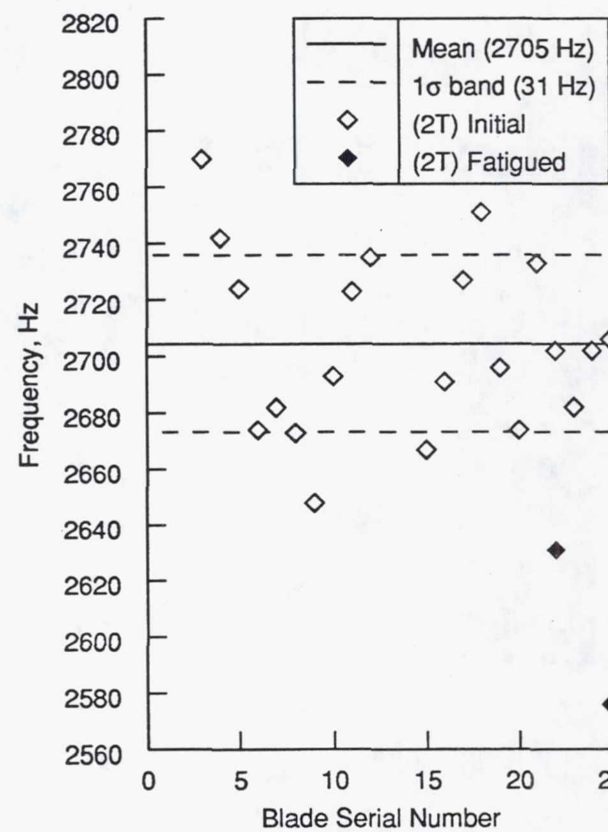

Figure 7 – CM1DF second torsion mode

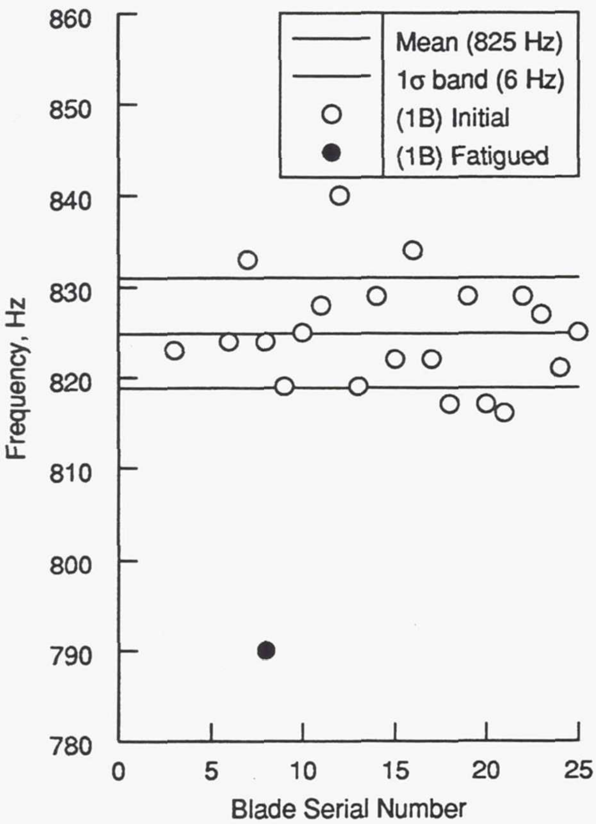

Figure 8 – CM1DA first bending mode

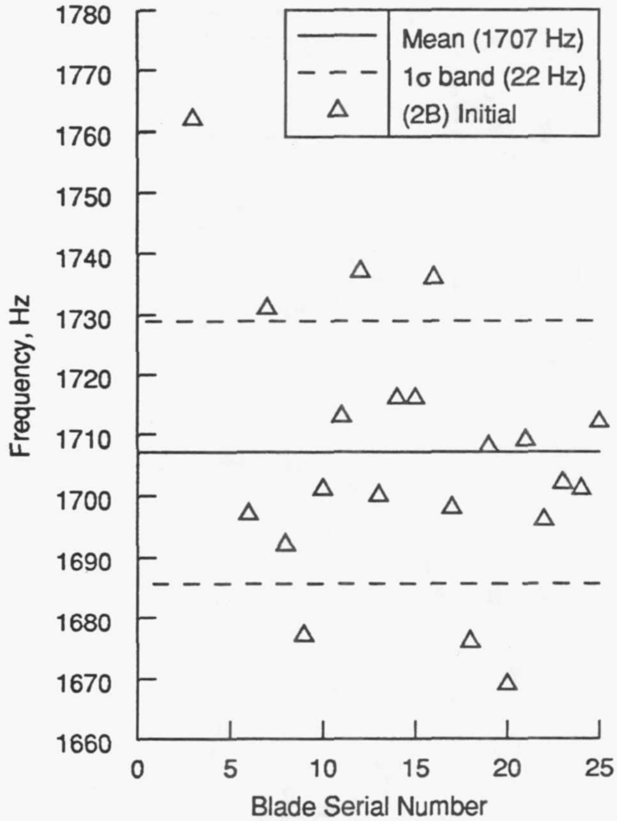

Figure 9 – CM1DA second bending mode

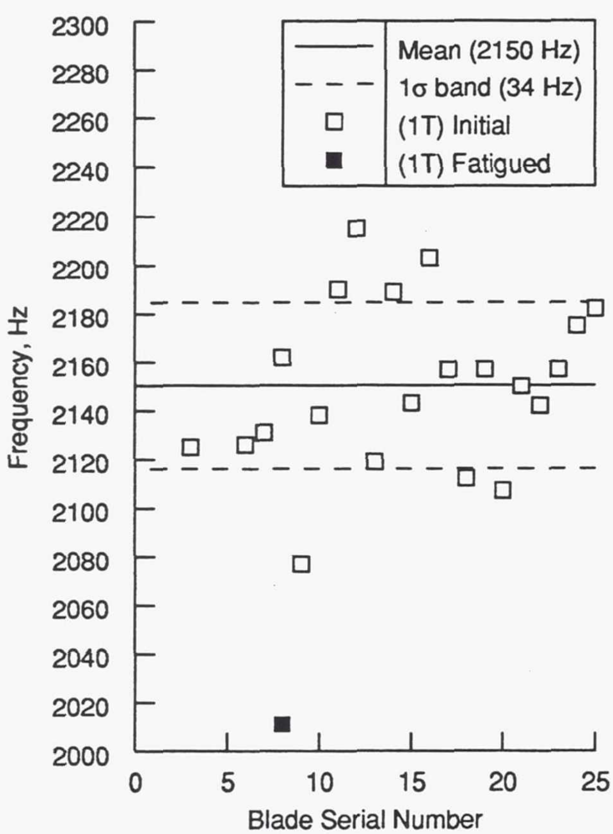

Figure 10 – CM1DA first torsion mode

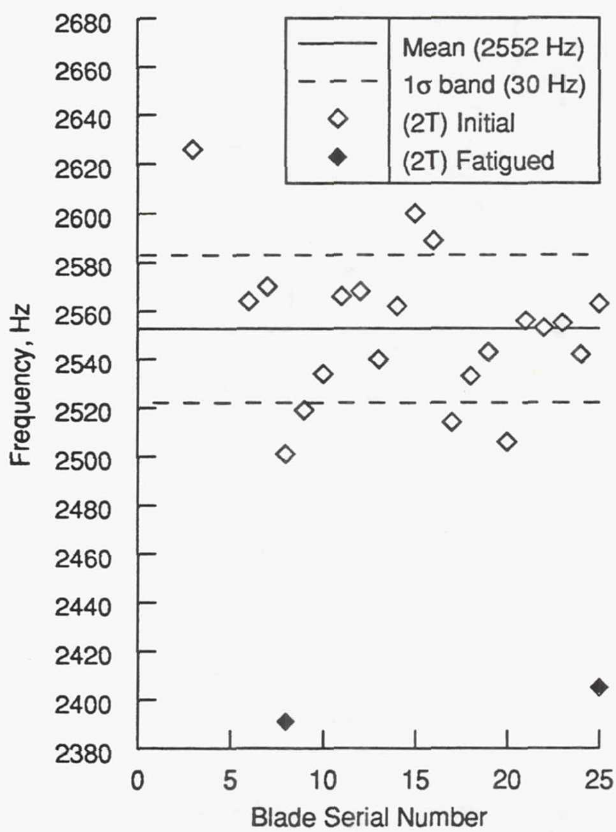

Figure 11 – CM1DA second torsion mode

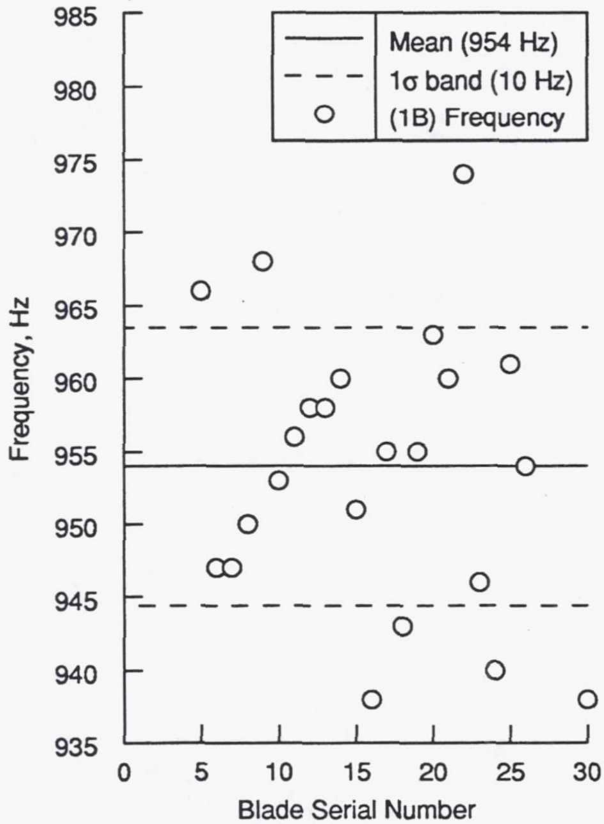

Figure 12 – CM2DF first bending mode

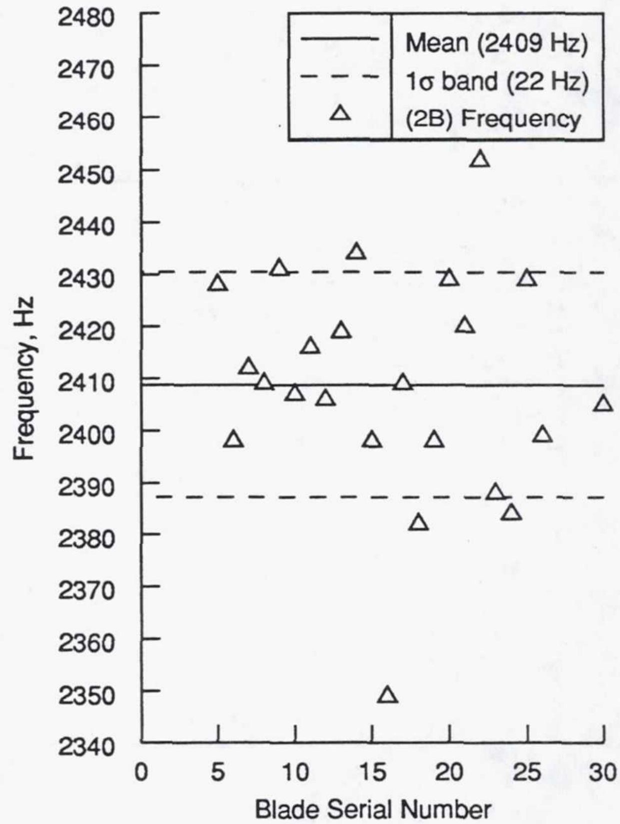

Figure 13 – CM2DF second bending mode

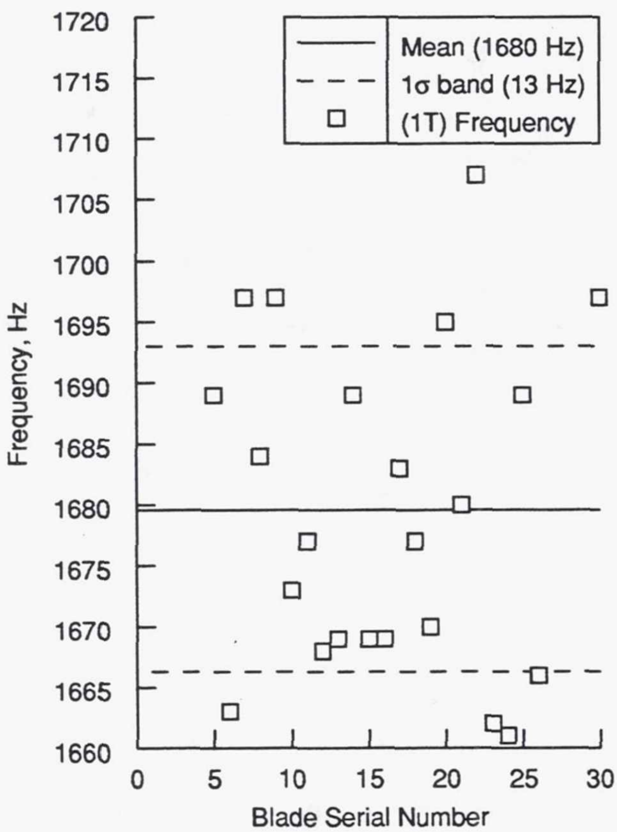

Figure 14 – CM2DF first torsion mode

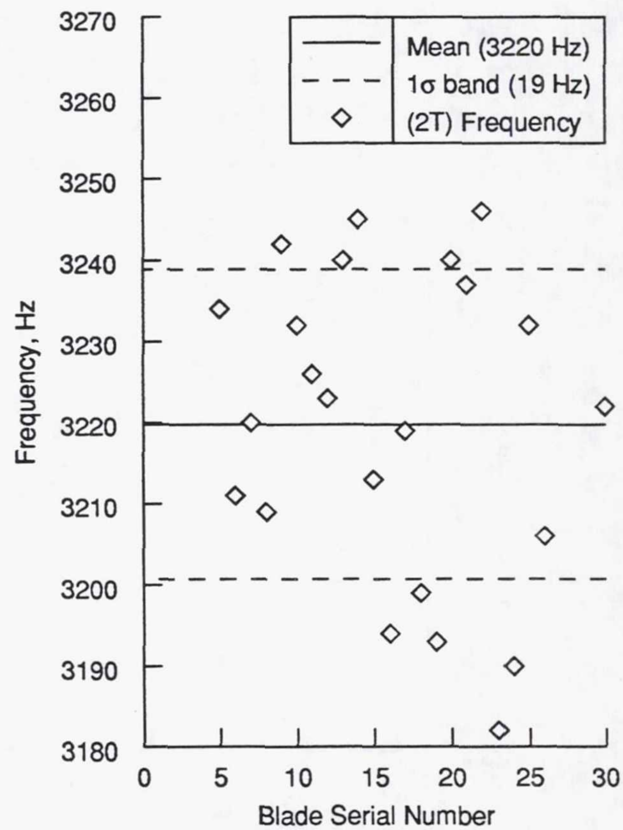

Figure 15 – CM2DF second torsion mode

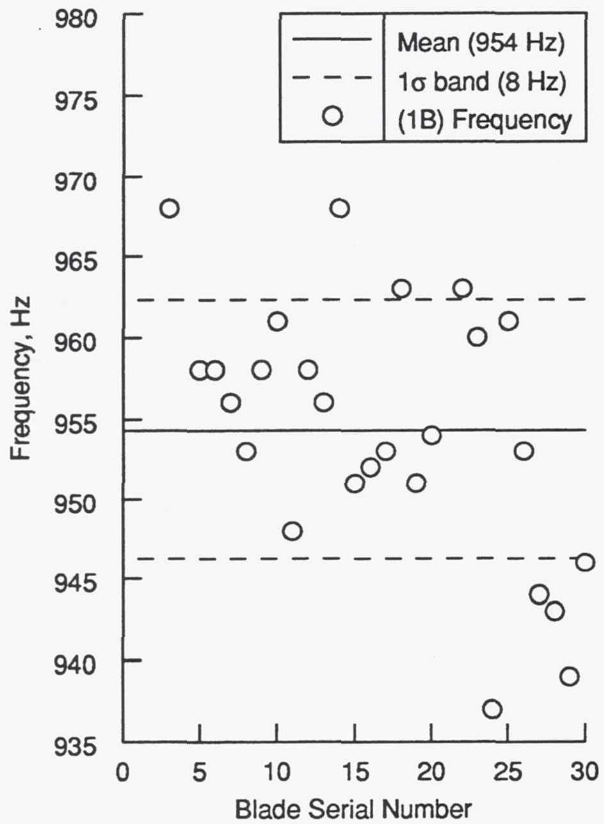

Figure 16 – CM2DA first bending mode

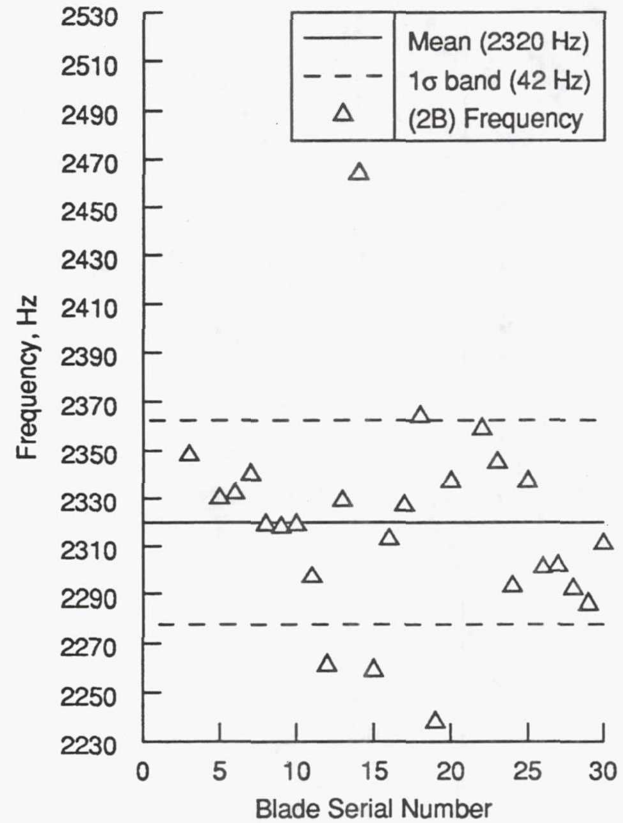

Figure 17 – CM2DA second bending mode

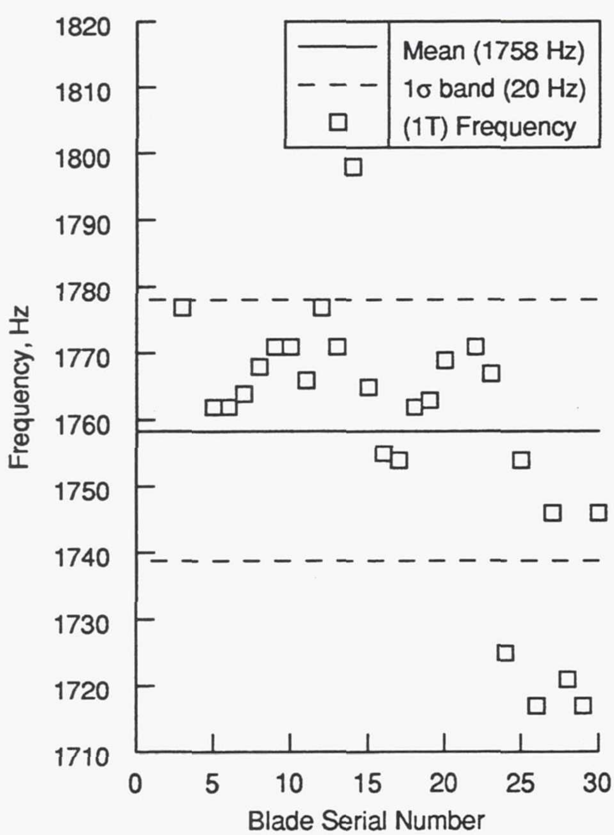

Figure 18 – CM2DA first torsion mode

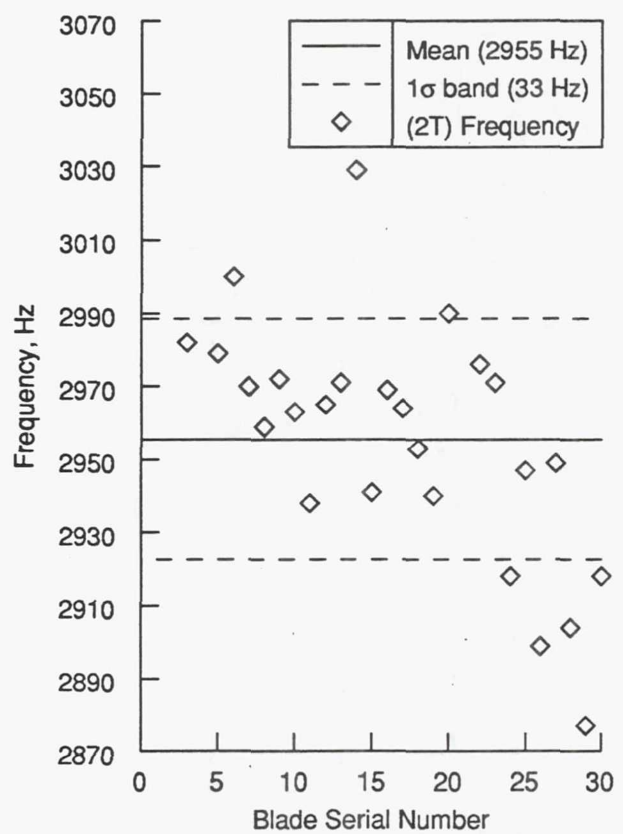

Figure 19 – CM2DA second torsion mode

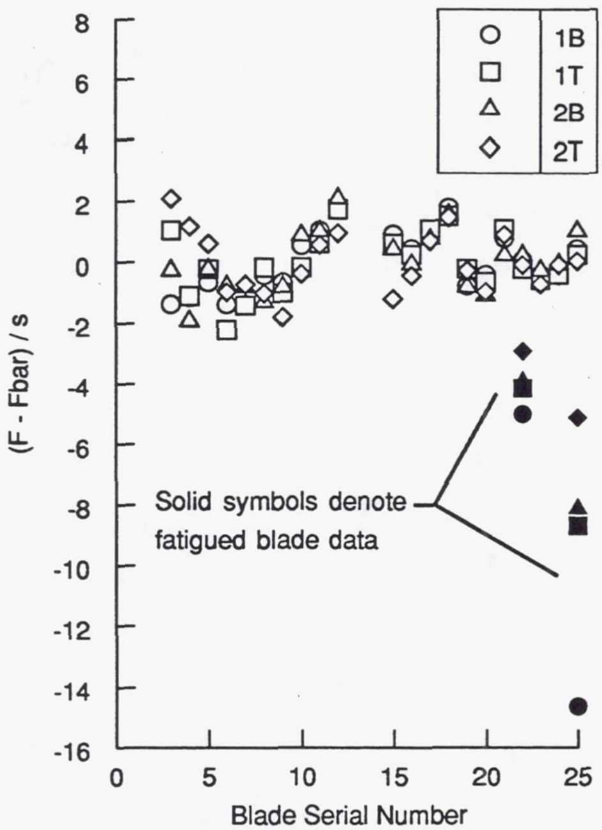

Figure 20 - CM1DF Frequency Errors

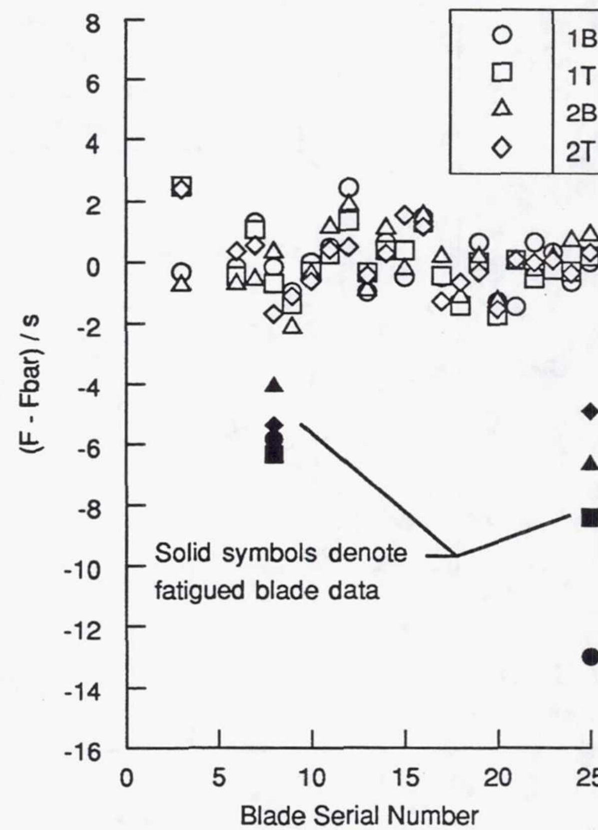

Figure 21 - CM1DA Frequency Errors

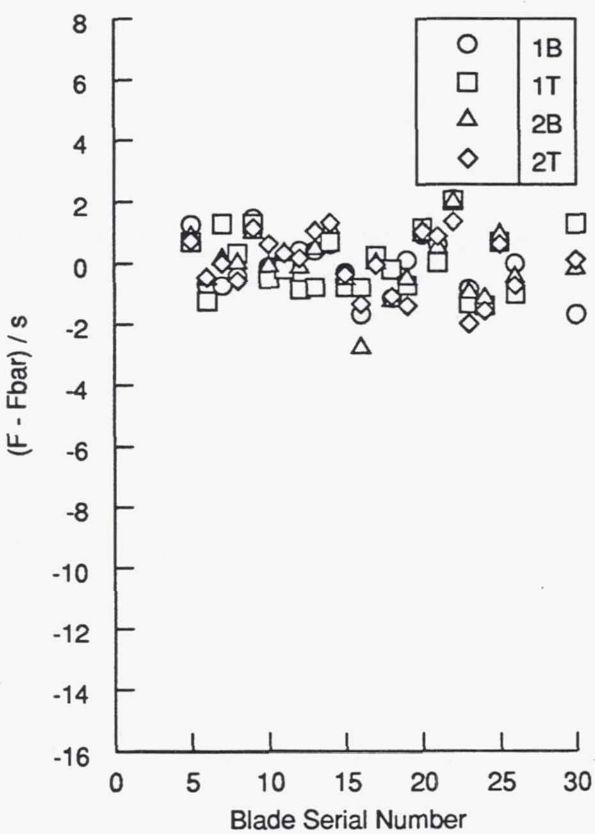

Figure 22 - CM2DF Frequency Errors

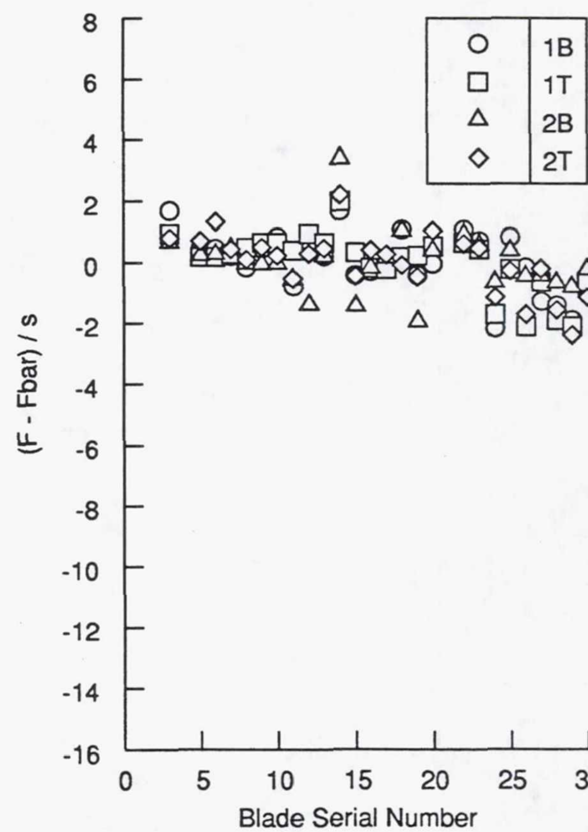

Figure 23 - CM2DA Frequency Errors

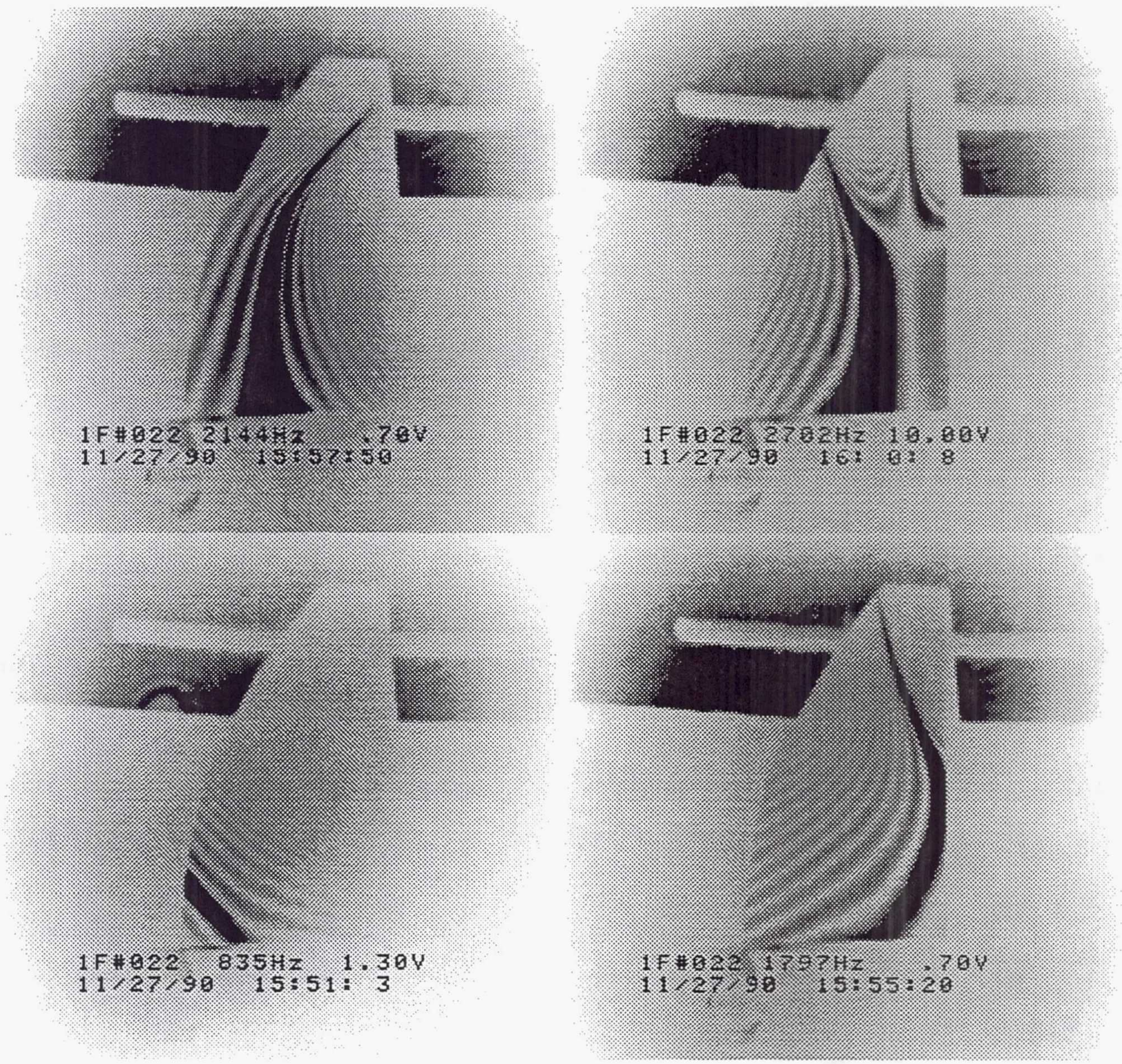

Figure 24 - CM1DF022 Four Hologram (1B,2B,1T,2T) Composite Image

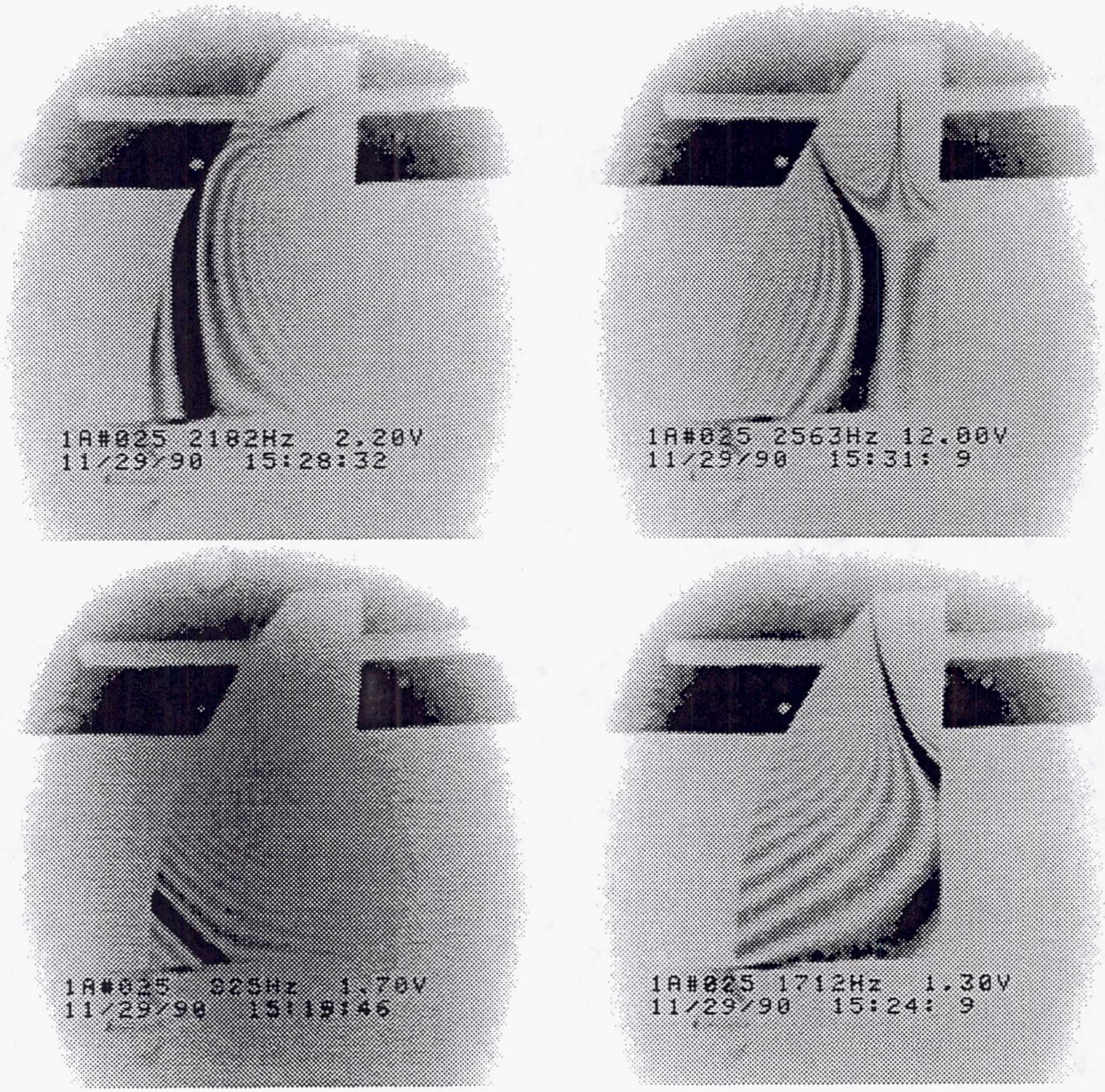

Figure 25 - CM1DA025 Four Hologram (1B,2B,1T,2T) Composite Image

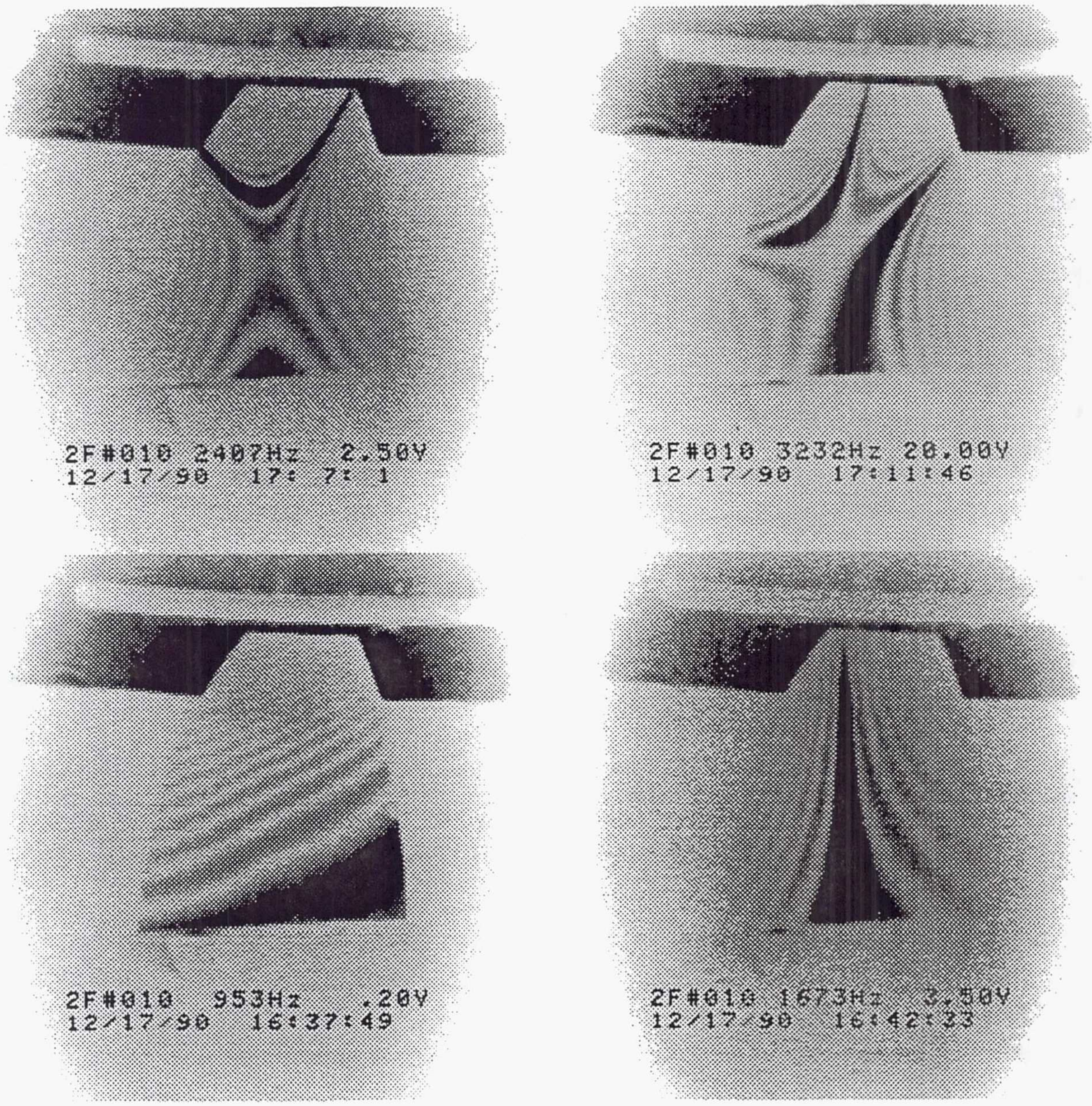

Figure 26 – CM2DF010 Four Hologram (1B,1T,2B,2T) Composite Image

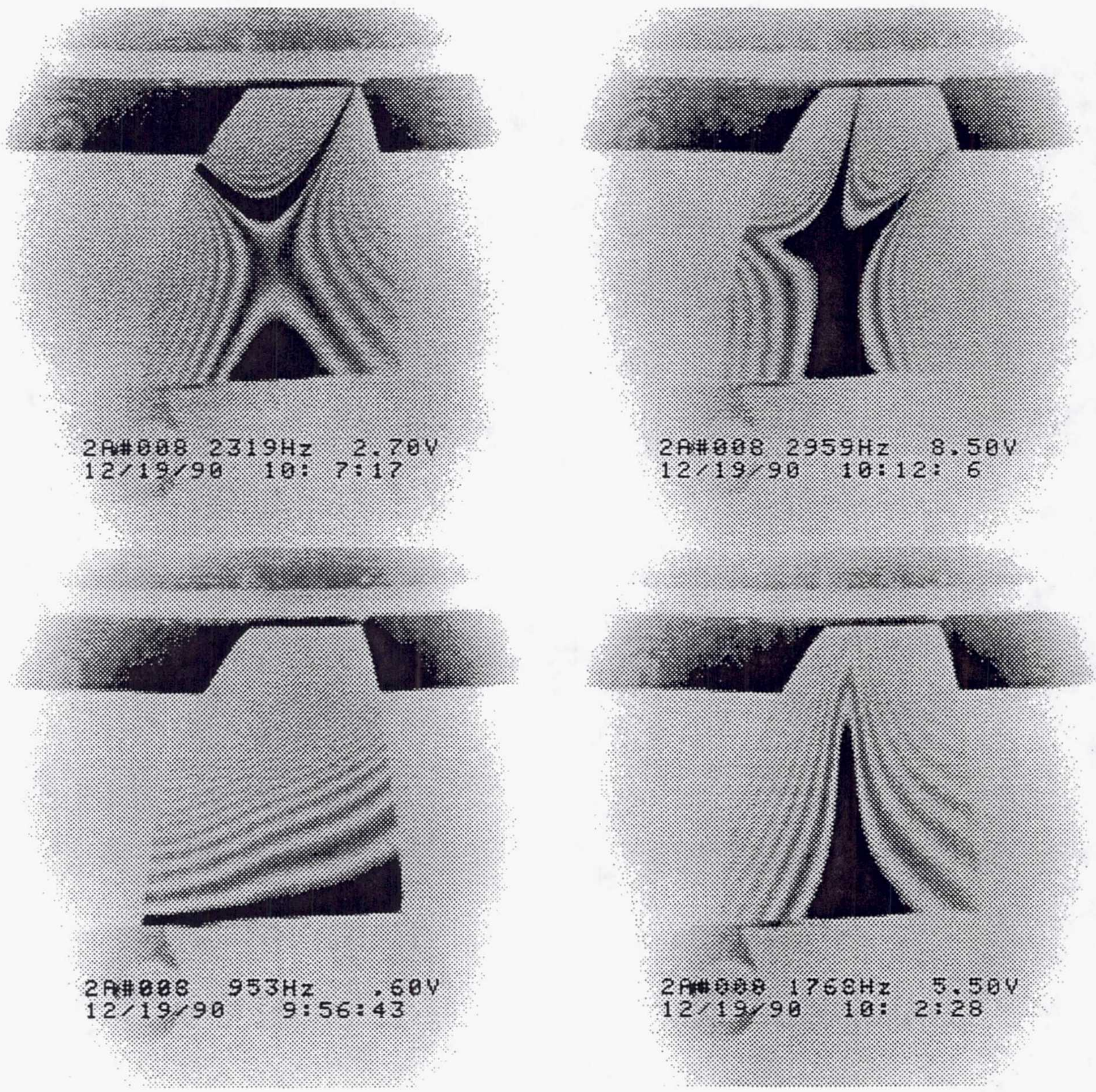

Figure 27 – CM2DA008 Four Hologram (1B,1T,2B,2T) Composite Image

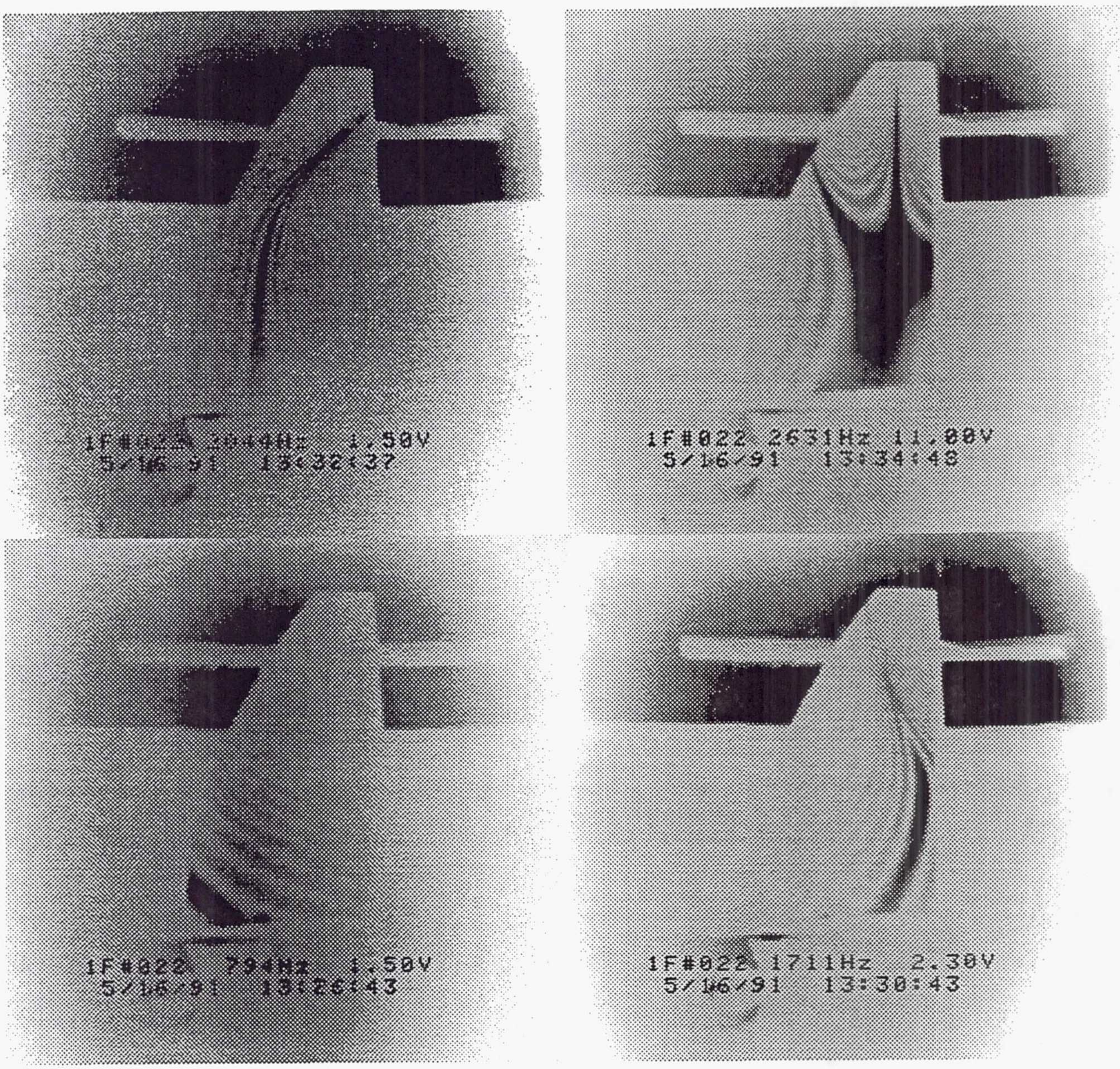

Figure 28 - Fatigued CM1DF022 Four Hologram Composite Image

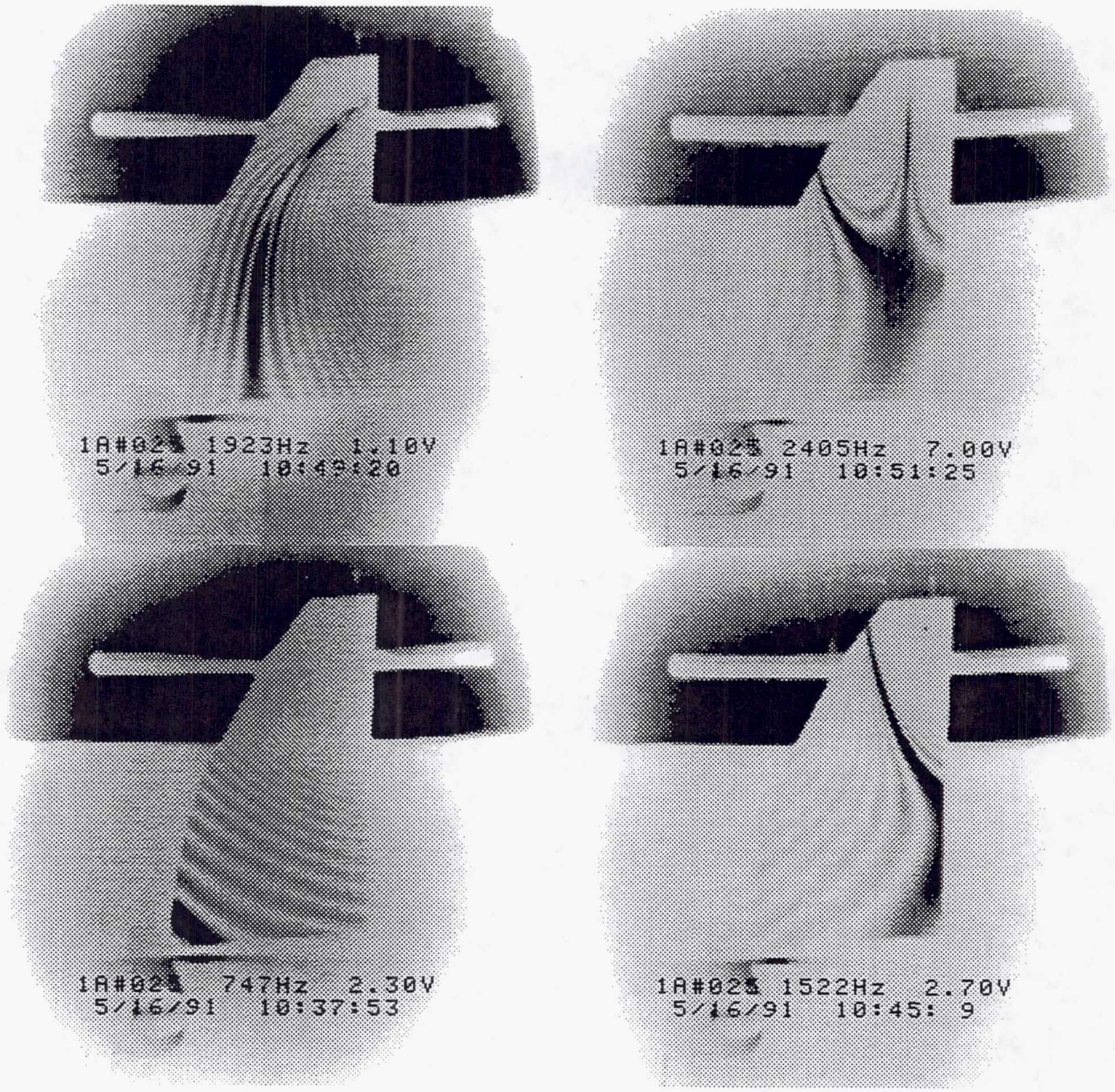

Figure 29 – Fatigued CM1DA025 Four Hologram Composite Image

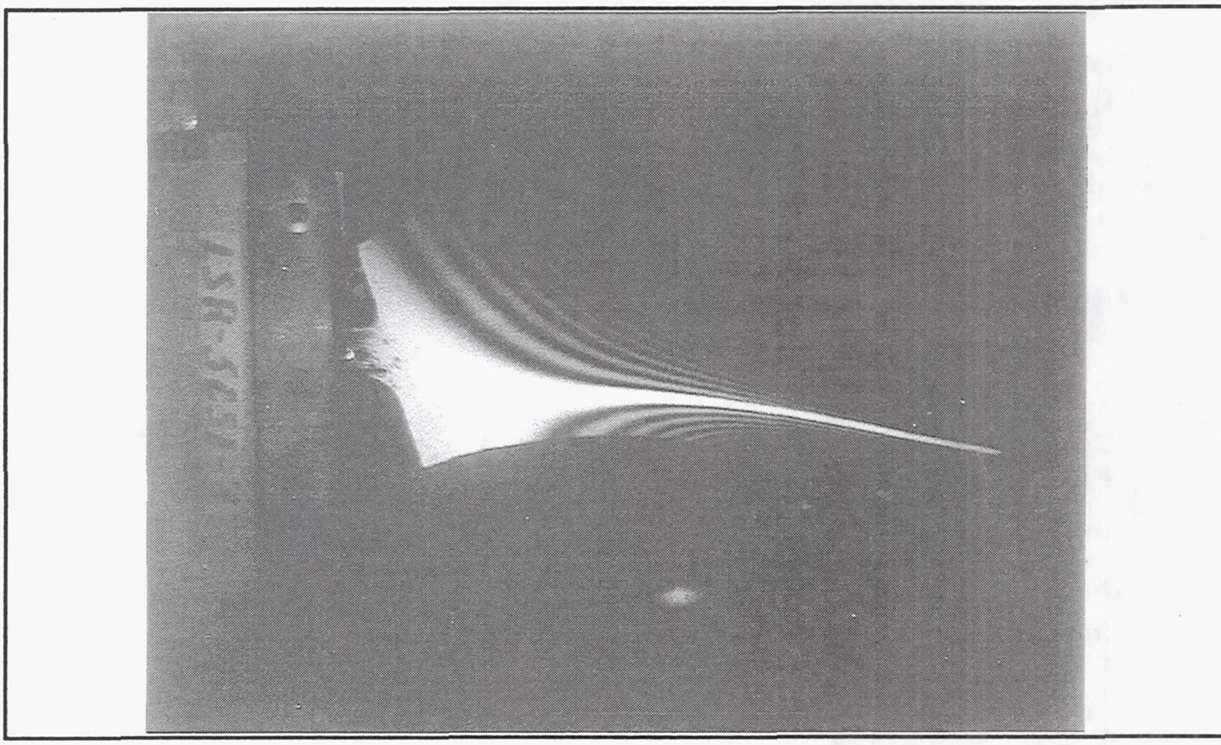

Figure 30 – Representative Photographic Hologram

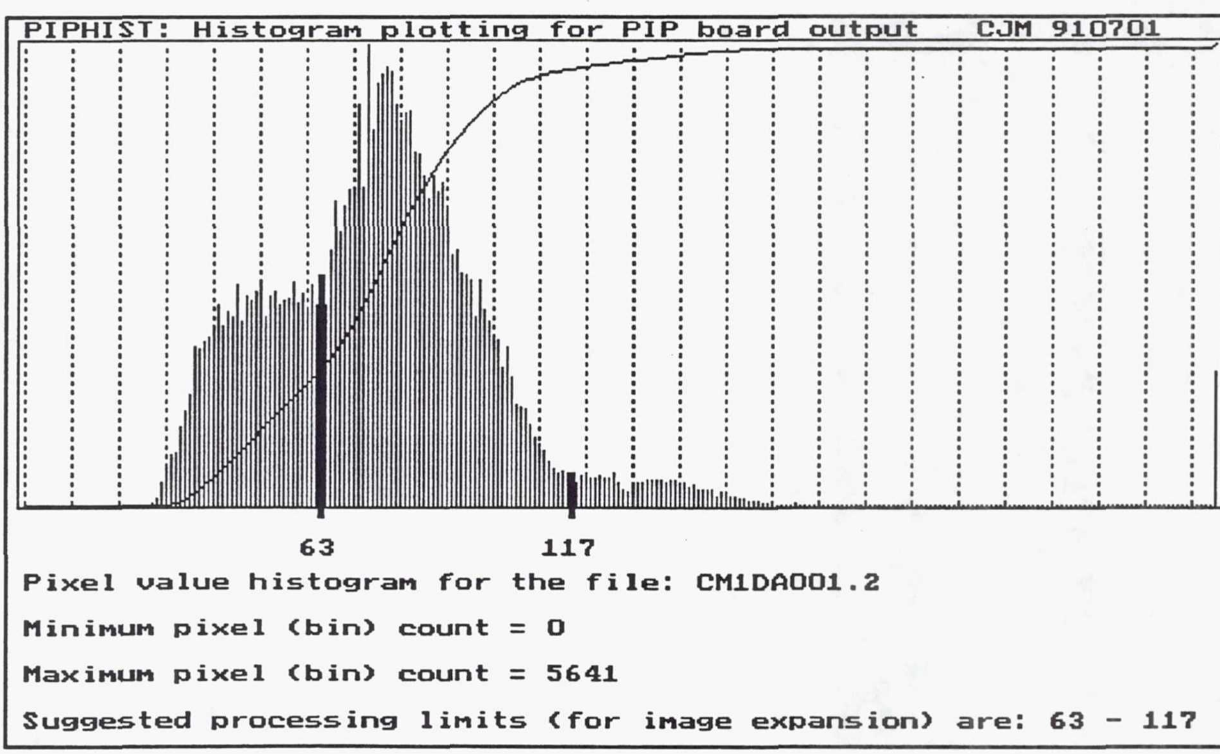

Figure 31 – PC Histogram display showing recommended image enhancement limits

Figure 32 – PC Histogram display showing recommended image enhancement limits

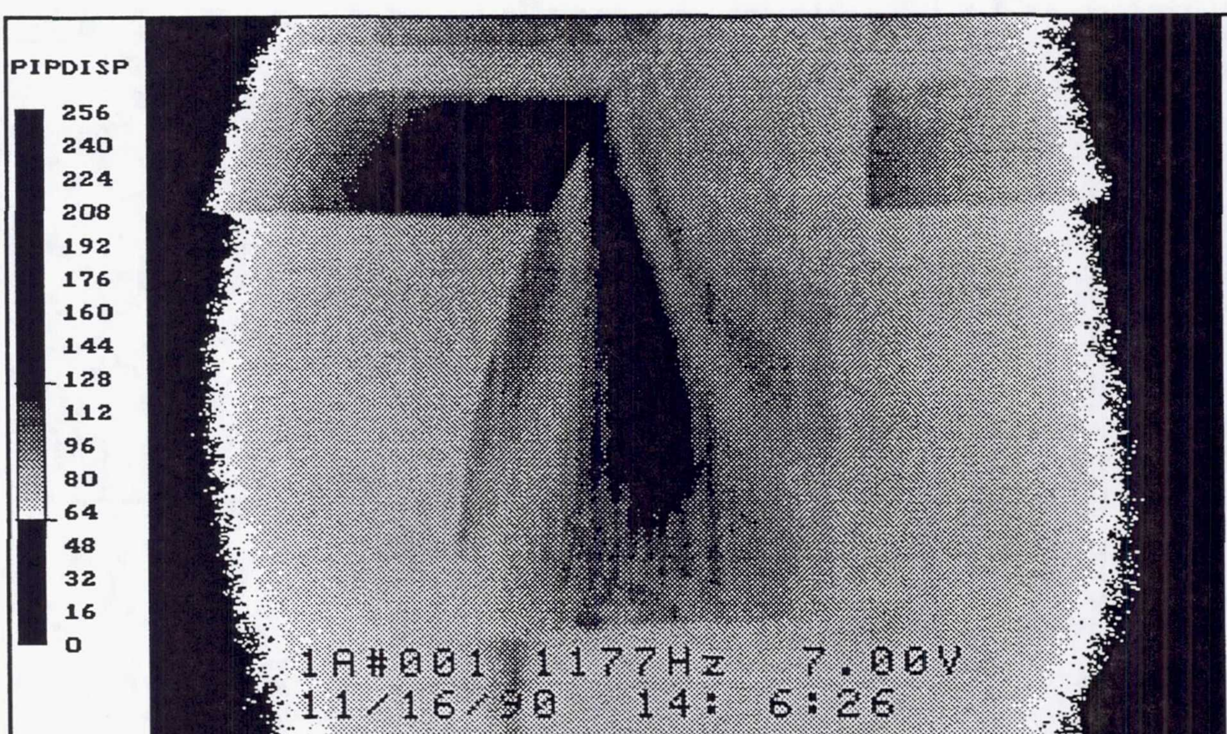

Figure 33 – PC Histogram display showing recommended image enhancement limits

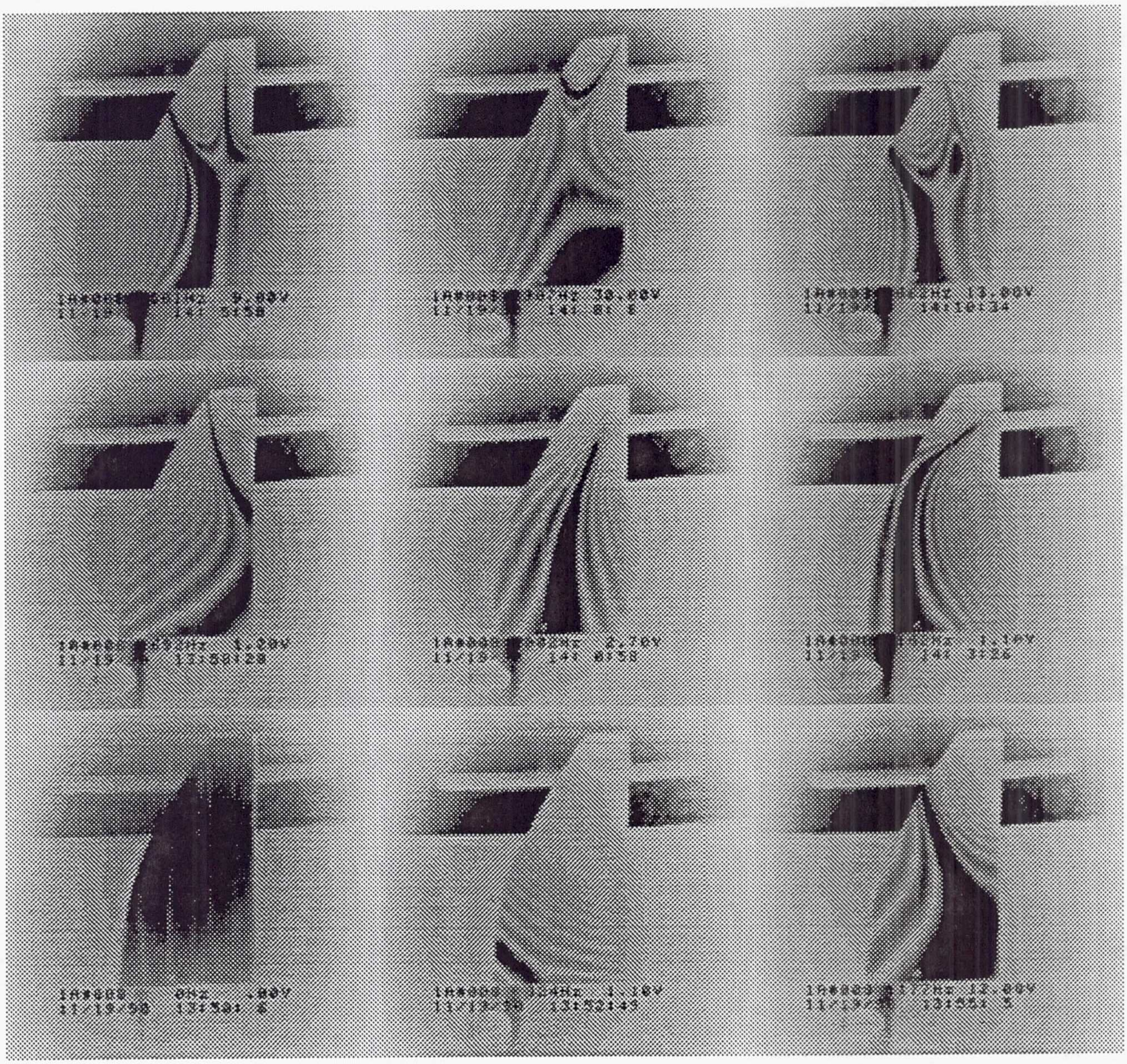

Figure 34 – Composite: static image and holograms of first 8 modes

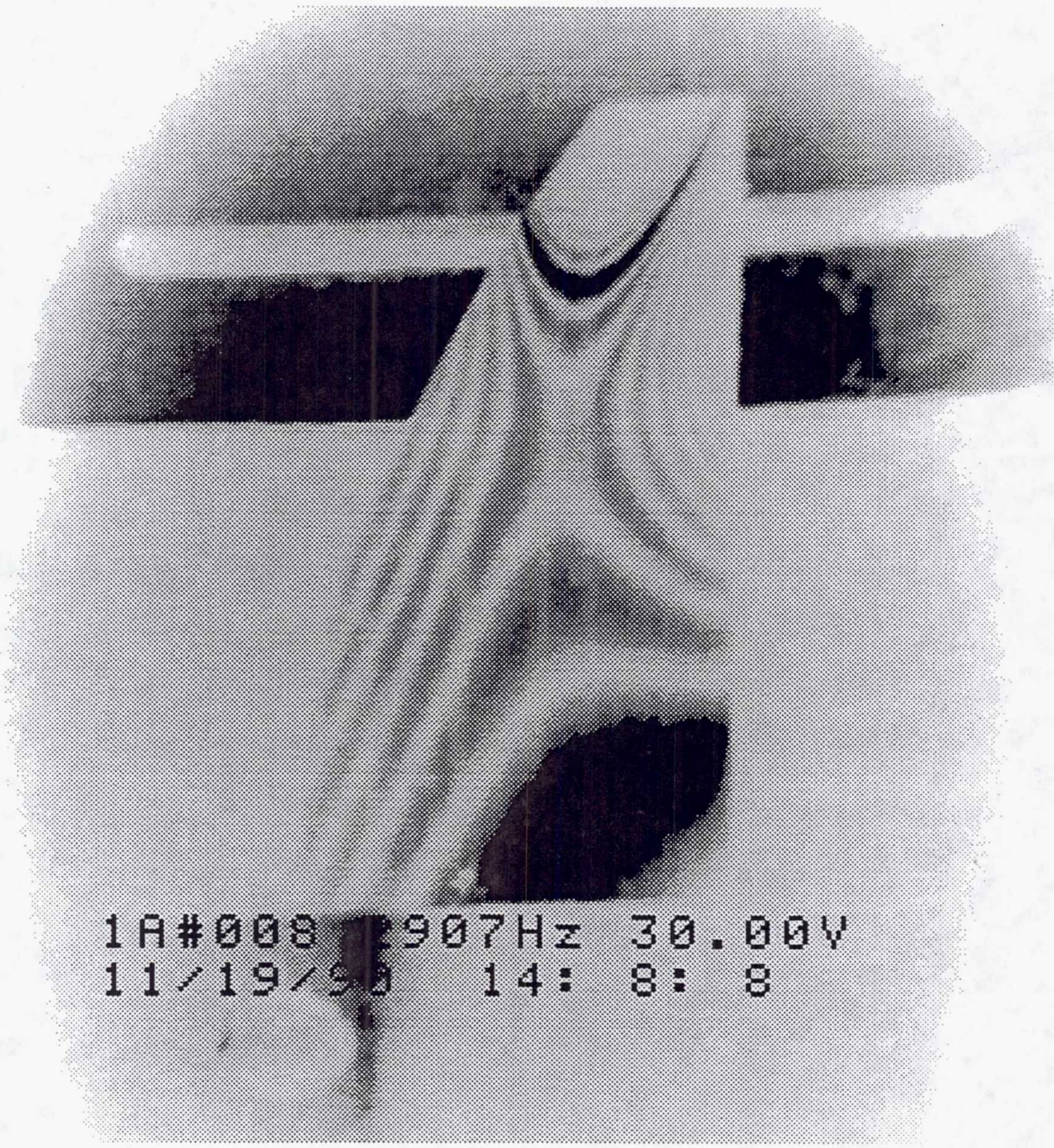

Figure 35 – Single mode hologram (uncompressed) printed in 16 gray levels

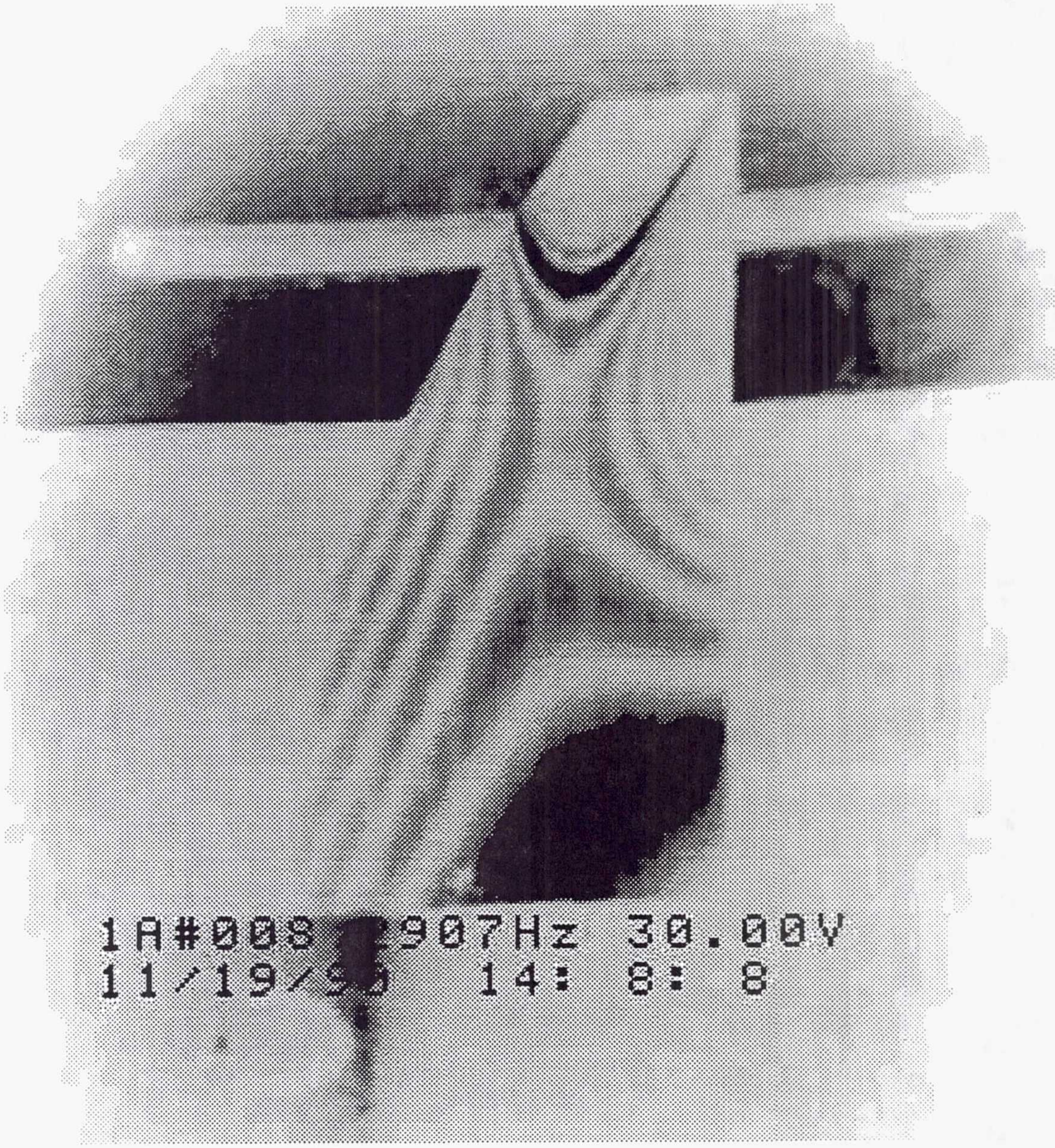

Figure 36 – Single mode hologram, JPEG compressed with Q=20, printed in 16 gray levels

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1994	Technical Memorandum

4. TITLE AND SUBTITLE

Holographic Testing of Composite Propfans for a Cruise Missle Wind Tunnel Model

5. FUNDING NUMBERS

WU–535–03–10

6. AUTHOR(S)

Christopher J. Miller

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–8203

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM–105271

11. SUPPLEMENTARY NOTES

Responsible person, Christopher J. Miller, (216) 433–6179.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 07

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

Each of the approximately 90 composite propfan blades constructed for a 55 percent scale cruise missile wind tunnel model were holographically tested to obtain natural frequencies and mode shapes. These data were used not only for quality assurance, but also to select sets of similar blades for each blade row. Presented along with the natural frequency data is a description of a computer-based image processing system developed to supplement the photographic-based system for holographic image analysis and storage. The new system is quicker and cheaper, the holograms are indexed better, and several engineers can access the data simultaneously. The only negative effect is a slight reduction in image resolution, which does not influence the end use.

14. SUBJECT TERMS

Holography; Image processing; File compression

15. NUMBER OF PAGES

43

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	