LA-SUB--96-64

AERODYNAMICS OF THE LARGE-VOLUME, FLOW-THROUGH DETECTOR SYSTEM

Final Report on Contract 448AJ0004-3Y

with

Los Alamos National Laboratory
Chemical Science and Technology Division
Mail Stop J-586
Los Alamos, New Mexico 87545

MARCH 1996

Submitted by

H. Reed, W. Saric, and D. Laananen (Principal Investigators)
and
C. Martinez, R. Carrillo, J. Myers, (Students) and D. Clevenger (Technician)

Aerospace Research Center
College of Engineering and Applied Science
Arizona State University
Tempe, AZ 85287-8006

ABSTRACT

The Large-Volume Flow-Through Detector System (LVFTDS) was specially designed by LNL to monitor alpha radiation emitted by plutonium, uranium, and americium in mixed-waste incinerator offgases; however, it can be adapted to other important monitoring uses that span a number of potential markets, including site remediation, indoor air quality, radon testing, and mine shaft monitoring. The goal of this effort was to provide mechanical-design information for the installation of the LVFTDS in an incinerator, with a particular emphasis on the ability to withstand the high temperatures and high flow rates expected. The work was successfully carried out in three stages: (1) Calculation of the pressure drop through the system; (2) Materials testing to determine surrogate materials for wind-tunnel testing; (3) Wind-tunnel testing of an actual configuration.

TABLE OF CONTENTS

1 INTRODUCTION

Incineration is an attractive means for disposing transuranic wastes because 1) the volume of material is greatly reduced, 2) hazardous materials are converted to nonhazardous waste gases, and 3) the resulting residue meets Federal requirements without the need for further processing. However, the incinerators must not be allowed to emit radioactive particles. There is therefore a need for real-time, on-line monitoring of incinerators and other stacks. In its mechanical design and installation, the detector system should provide a fast-response alert and system shut-down when even a low level of radiation is sensed.

The Large-Volume Flow-Through Detector System (LVFTDS) was specially designed to monitor alpha radiation emitted by plutonium, uranium, and americium in mixed-waste incinerator offgases; however, it can be adapted to other important monitoring uses that span a number of potential markets, including site remediation, indoor air quality, radon testing, and mine shaft monitoring. This new technology is much more sensitive than current monitors and, by monitoring large volumes of air, eliminates the obvious problems associated with sampling.

The goal of this effort is to provide mechanical-design information for the installation of the LVFTDS in an incinerator, with a particular emphasis on the ability to withstand the high temperatures and high flow rates expected.

2 COMPUTATIONAL AERODYNAMICS - STAGE 1

The work in this section was done by Mr. Christopher Martinez under the supervision of Dr. Helen Reed.

The computational stage involved a series of simple spreadsheet models as well as complex-C-language programs to analyze both the flow behavior within a detecting cell (e.g. pressure drop) and the influence of the flow on the walls of the detector (e.g. shear stress). The input variables to the computations are as follows:

Detector cell length in the streamwise direction x

Temperature (ambient)

Pressure (ambient)

Uniform inlet flow speed (= U_O)

Distance between detector plates (= L)

Number of intervals along the streamwise direction x

Number of inlet Reynolds numbers $Re_O = U_O L/\nu$

Viscosity is computed by Sutherland's formula, and pressure, temperature, and fluid density are related by the ideal gas law. The following ratios are defined:

alpha = streamwise distance from the inlet divided by the distance between the detector plates

beta = local centerline speed U divided by the uniform inlet speed

gamma = local displacement thickness divided by the distance between the detector plates

Considering an entrance-flow formulation and applying basic boundary-layer theory, the local Reynolds number $Re = Ux/\nu$, displacement thickness, and local centerline speed are computed. From these results, wall shear stress and pressure drop are determined. The length along x at which laminar flow becomes turbulent is assumed to occur at a local Reynolds number Re of 250,000. Results for various detector configurations are shown in Appendix A. Upon completion of the wind-tunnel tests conducted in Stage 3, the flow was shown to be laminar throughout. The computations based on this result agree exactly with the experiment.

3 MECHANICAL PROPERTY MEASUREMENT - STAGE 2

The work in this section was done by Mr. Christopher Martinez under the supervision of Dr. David Laananen.

For use in the wind tunnel model, it was necessary to select a low-cost substitute for the scintillator material. In order to match the dynamic response characteristics, the material to be used in the model would have to possess similar elastic properties and density. Although the density of BC-434 and BC-438 (1.05 g/cm^3) was available, the Young's modulus had to be determined by testing.

In order to produce ASTM-standard tensile test specimens with the dimensions shown in Figure 3.1, four 0.5-in. x 9.0-in. (12.7 mm x 229-mm) strips were cut from a sheet of BC-434. To avoid premature fracture in the serrated jays of the testing machine, compliant end tabs 1/8 in. (3.2 mm) thick were bonded to the BC-434 strips. A 350-ohm biaxial resistance strain gage was bonded to one side of each specimen.

Tensile tests were performed using a 10,000-lb Instron testing machine in the College of Engineering and Applied Sciences Mechanical Testing Laboratory. Two specimens fractured prematurely at very low tensile load, probably due to some small defects introduced during specimen preparation. These failures demonstrated the extremely brittle nature of the material, and great care was exercised in handling the remaining specimens.

The specimens reached a maximum tensile load at failure between 89 lb and 108 lb. Because the sensitivity of the load cell in this low range presented a possible problem, it was decided to verify the results with bending tests. Specimens were loaded in three-point bending with a standard 10-lb weight, and deflection at mid-span was measured. The tests were repeated for BC-438 and the modulus of elasticity for each material was calculated as described below.

For beam of length *L*, simply supported at both ends and subjected to concentrated load *P* at mid-span, the maximum deflection δ is given by

$$\delta = \frac{PL^3}{48EI}$$

in which *E* is the modulus of elasticity and *I* is the moment of inertia of the beam cross section with respect to a transverse axis at the centroid. This equation is then solved for the modulus of elasticity.

The moment of inertia of the specimen cross section is given by

$$I = \frac{bh^3}{12}$$

where *b* is the width and *h* is the thickness.

The average Young's modulus values for the specimens tested were as follows:

BC-434:	4.2×10^5 lb/in.^{2} (2.9 GPa)
BC-438:	4.5×10^5 lb/in.^{2} (3.1 GPa)

Because the moduli for the two materials are nearly the same, it was decided to locate for use in the wind tunnel model a substitute material whose Young's modulus lay between the above measured values. The material selected was a methacrylate that was available in the desired thickness of 1 mm.

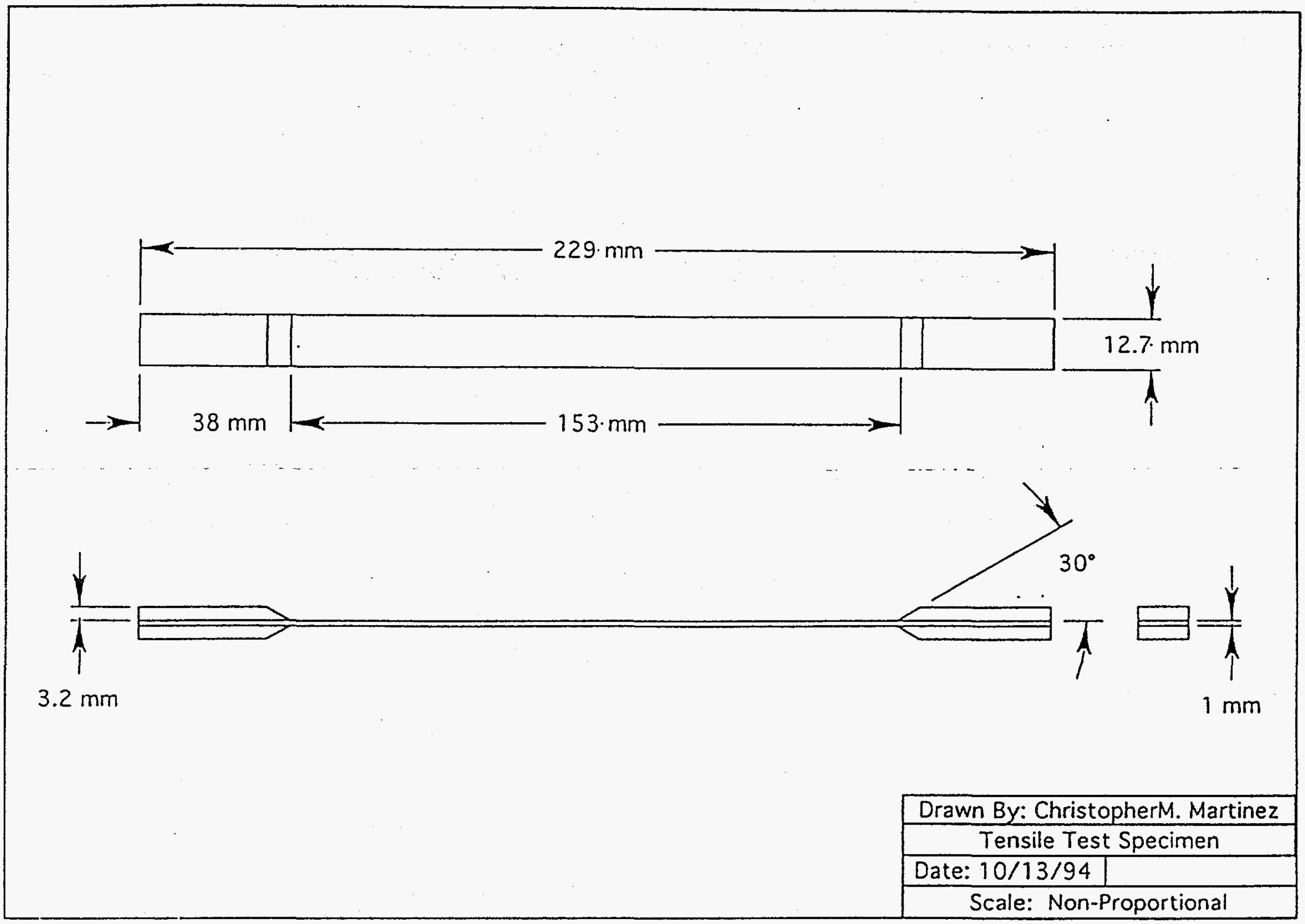
229 mm
12.7 mm
38 mm
153 mm
30°
3.2 mm
1 mm
Drawn By: ChristopherM. Martinez
Tensile Test Specimen
Date: 10/13/94
Scale: Non-Proportional

Figure 3.1.

4 WIND-TUNNEL MEASUREMENTS AND MODEL DEVELOPMENT - STAGE 3

The work at this stage was initiated about the same time that Mr. Martinez transferred out of the College of Engineering. The test-section design work was done by Mr. Martinez under the supervision of Dr. William Saric. The test-section fabrication and the wind-tunnel testing was done by Mr. Ruben Carrillo (MAE Graduate Student), Mr. Joseph Myers (MAE Undergraduate Student), and Mr. Danny Clevenger (Wind Tunnel Technician assigned to the ASU Unsteady Wind Tunnel) under the supervision of Dr. Saric.

4.1 Experimental Facility

The LVFTDS experiment is performed in the ASU Flow-Visualization Tunnel. The facility is a low-speed, open-circuit tunnel with a unique octagon cross section. The tunnel was completely designed and built at Arizona State University with flow quality and experiment versatility as design drivers. The tunnel is driven by a 10 hp variable-speed AC motor and a single-stage axial fan. The fan has four adjustable blades, and the diameter of the fan housing is 0.76 m. A variable-frequency drive interfaces to the motor to control fan rotation. For the test section used in this experiment, speeds are continuously variable and accurately controlled between 3 m/s and 42 m/s.

The Flow-Visualization Tunnel, shown in Figure 4.1, has several features designed to produce low turbulence levels. The unique octagon shape of the facility reduces the effects of corner vortices while honeycomb and screens reduce flow fluctuations. The contraction cone is designed using a fifth-order polynomial to eliminate curvature discontinuities at the ends. The area contraction is 25:1 over 2.84 m. To minimize vibrations, the test section is isolated from the diffuser and fan section by a flexible coupling. The fan section is also isolated from its frame and the diffuser with a flexible coupling.

The facility's octagonal test section is 2.4 m long with a 0.21 m^2 cross-sectional area. The side panels are all Plexiglas for excellent visibility, and the test-section dimensions allow for a variety of experiments. The parallel sides of the test section allow for several viewing planes, and are well-suited for photography and laser-Doppler anemometry. Theatrical smoke is used to visualize the flow. The smoke can be injected at the inlet of the tunnel or at the settling chamber. The smoke generator incorporates a two-dimensional traverse for positioning the injected smoke.

4.2 Model

To accommodate the LVFTDS experiment, a new test section was designed for the Flow-Visualization Tunnel. To mate this test section with the Flow-Visualization Tunnel, an octagon-to-square contraction and a square-to-octagon diffuser were designed for placement before and after the LVFTDS test section, respectively as shown in Figure 4.2. The contraction section serves to change the cross section from an octagon (at the end of the tunnel's contraction cone) to a square (to suit the new test section) as shown in Figure 4.3. Similarly, the diffuser section mates the square test section to the tunnel's octagonal diffuser as shown in Figure 4.4.

Arizona State University
Flow-Visualization Tunnel

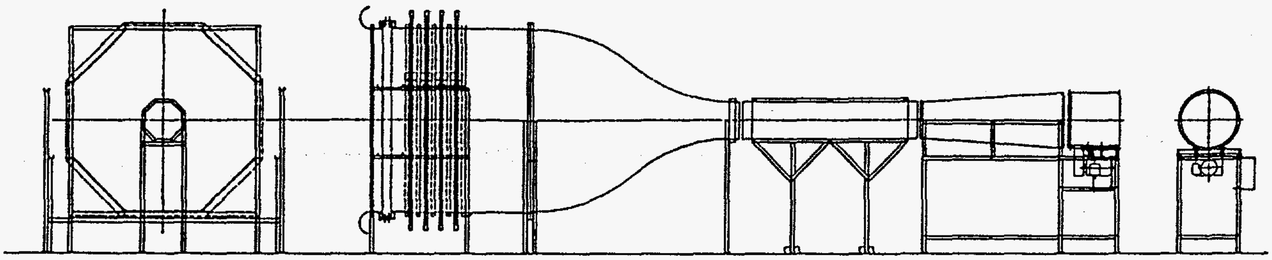

Figure 4.1.

Octagonal-to-Square Test Section
Designed for LVFTDS

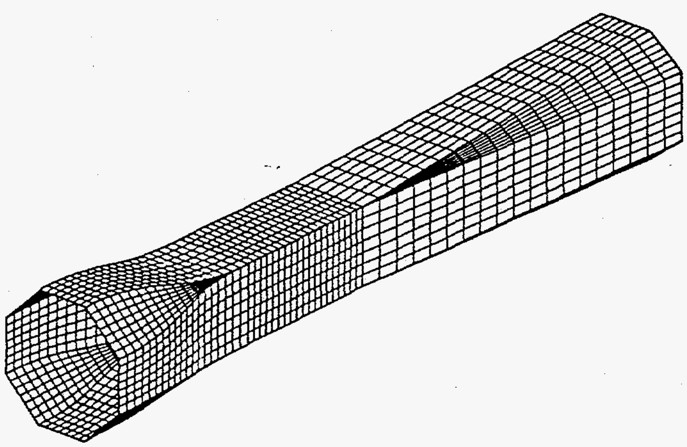

Figure 4.2.

LVFTDS Contraction Section

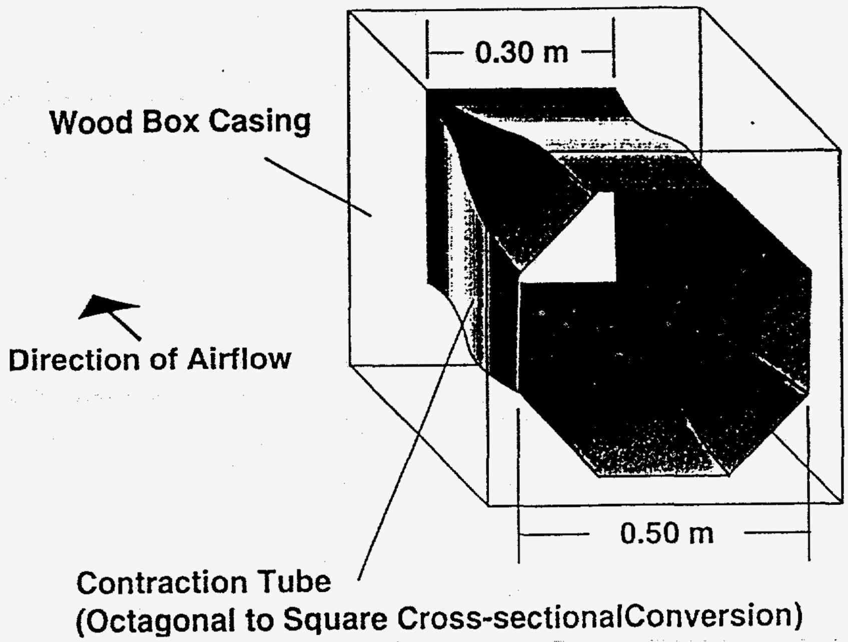

Figure 4.3.

LVFTDS Diffuser Section

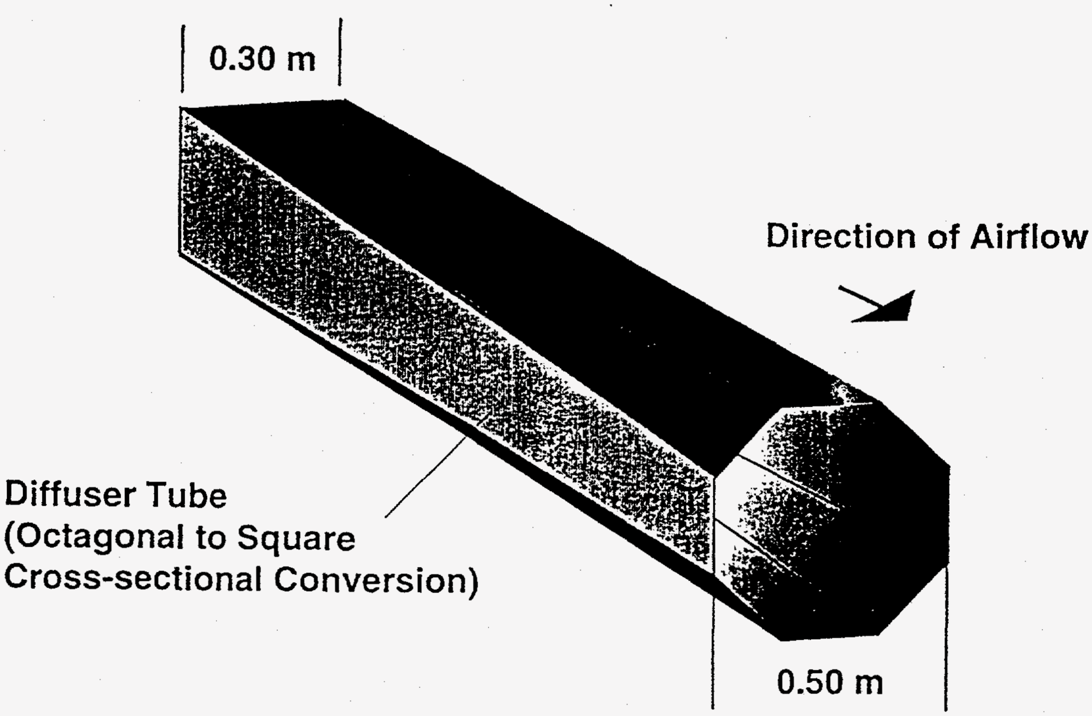

Figure 4.4.

This new test section has a square cross section with an area of 0.09 m^2. The test section allows for a full-scale (1-foot square) model of the LVFTDS, and it holds up to fourteen vertically-mounted test plates as shown in Figures 4.5 to 4.8. These plates are mounted in a groove-and-lock device that allows for quick and easy plate removal. All sides of the test section are Plexiglas to provide excellent visibility of the experiment.

4.3 Experimental Procedure

Preliminary tests of the LVFTDS show that airflow across the plates causes them to deflect from their initial parallel arrangement. The magnitudes of these observed deflections are directly proportional to the freestream velocity. To quantify the deflections, two characteristic velocities are defined. The first is the velocity at which the plates begin to diverge, and the second is the velocity at which the plates deflect over 5 mm. These velocities will be referred to as "divergence start" and "divergence limit", respectively and are recorded for each run condition.

To combat the deflections, a procedure is developed in which airfoil-shaped structural supports are attached to the leading and trailing edges of the LVFTDS plates. In the first case, the LVFTDS system is placed in the test section without any modifications. In the second case, a support is attached across the leading edges of the LVFTDS plates, centered between the upper and lower test-section walls. In the next case, a second airfoil-shaped support is centered across the trailing edges of the LVFTDS plates. For the fourth case, a total of four supports are attached to the plates, with two spaced evenly at the leading and trailing edges.

In the next part of the experiment, a turbulence-inducing grid is installed upstream of the octagon-to-square contraction to study the effects of higher turbulence levels on the characteristic divergence velocities. The four cases described above are repeated for this configuration. A fifth case is added in which a total of six structural supports are used, with three spaced evenly at the leading and trailing edges.

In the last part of the experiment, pressure drop measurements are taken across the LVFTDS at locations between each plate. These measurements are used to assess the flow uniformity across the test section.

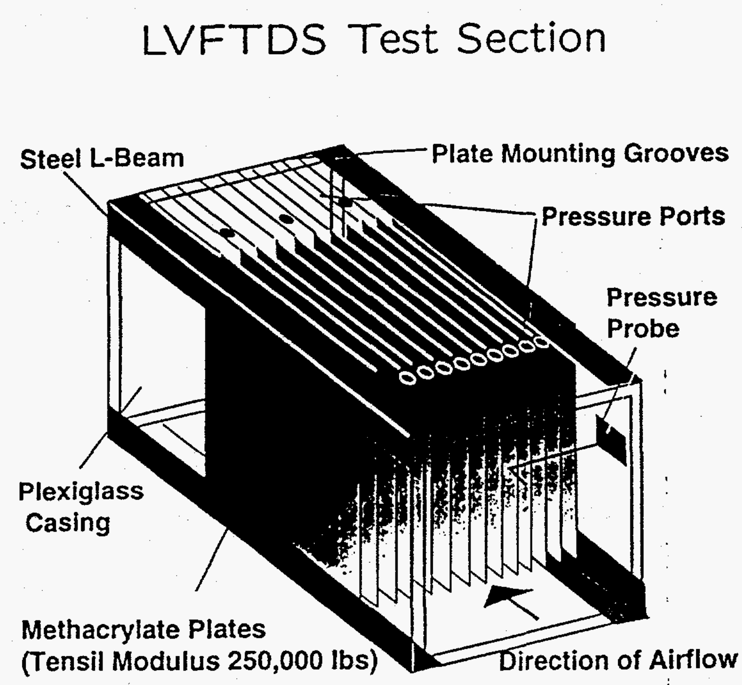

Figure 4.5.

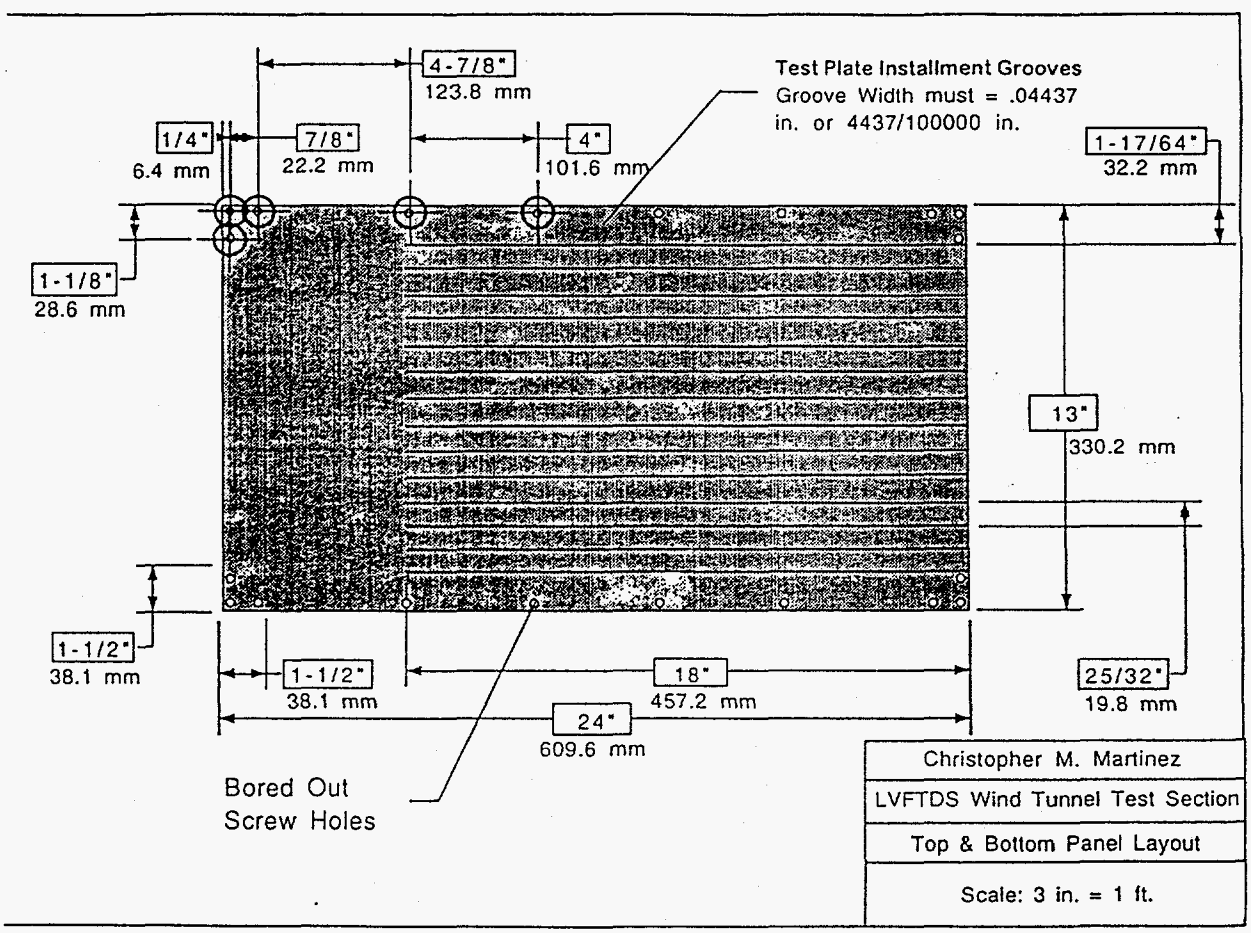

Figure 4.6.

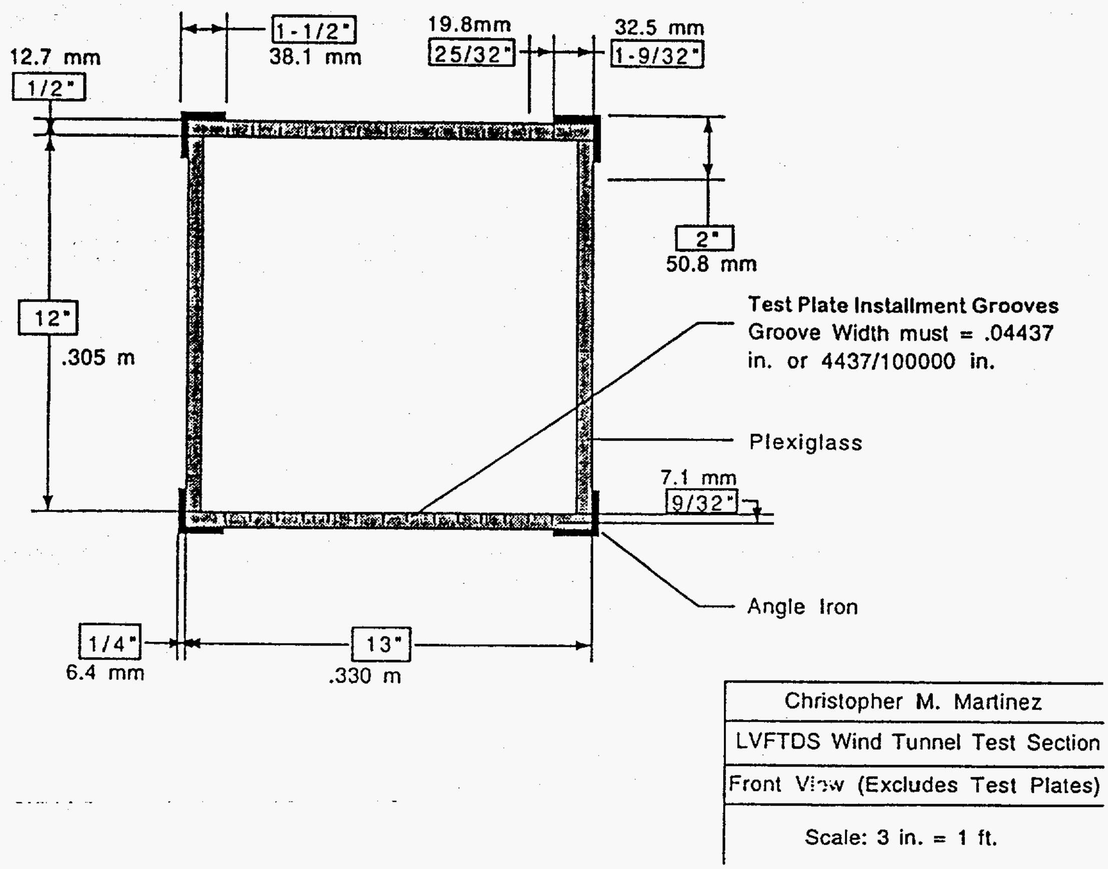

Figure 4.7.

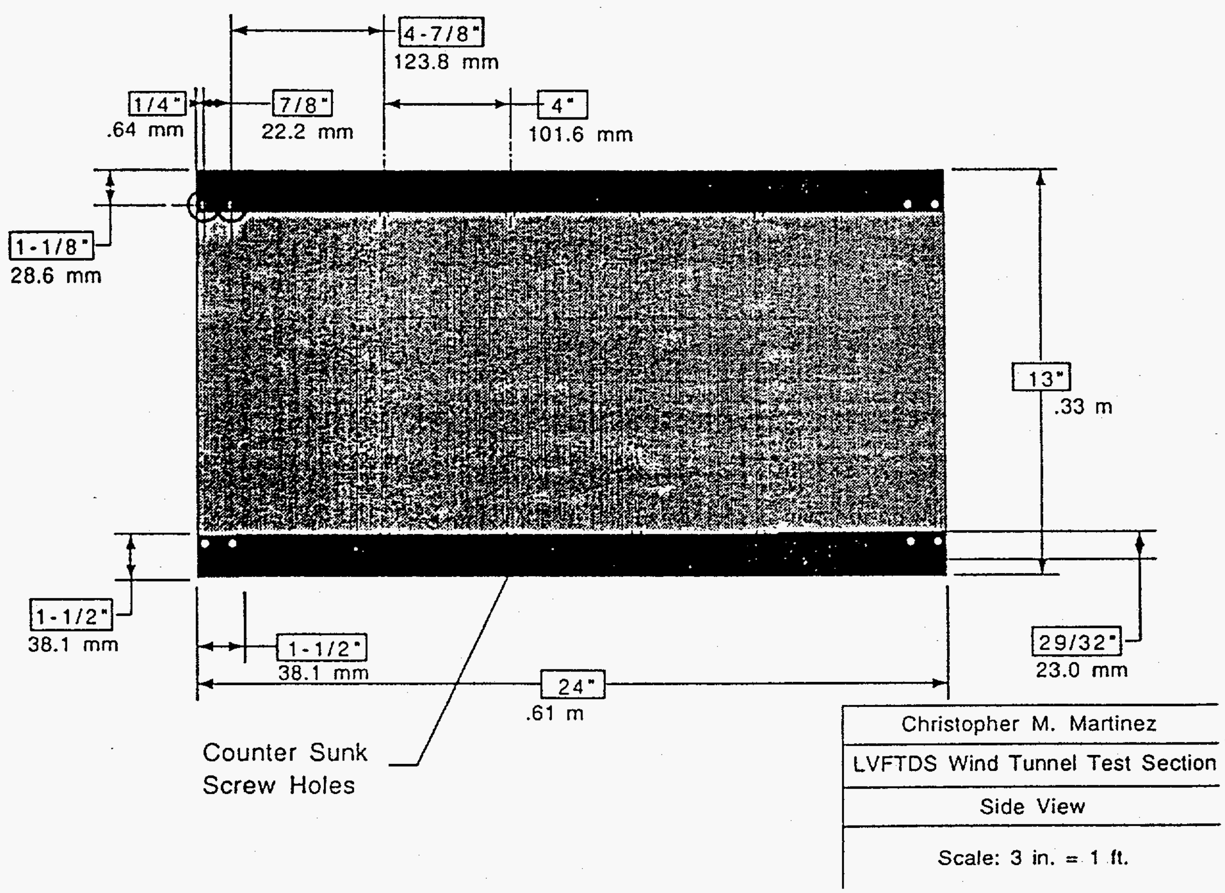

Figure 4.8.

4.4 Results

The LVFTDS experiment shows that the plates diverge from their initial parallel arrangement for all cases. As mentioned in the previous section, the magnitudes of the deflections are directly proportional to the freestream velocity. The addition of the airfoil-shaped structural supports, in general, increases the characteristic divergence velocities. The addition of the turbulence-inducing grid does not have a considerable effect on the LVFTDS.

The characteristic divergence velocities for the runs without the turbulence grid are presented in Table 4.1.

Number of Supports	Divergence-Start Velocity [m/s]	Divergence-Limit Velocity [m/s]
0	4.6	10.1
1	15.0	20.3
2	17.0	22.5
4	17.3	21.5

Table 4.1. Characteristic divergence velocities for runs without the turbulence grid.

These data are presented graphically in Figure 4.9. As seen in Table 4.1, deflections begin at relatively low freestream velocities when no structural supports are used. The addition of one support across the leading edges triples the divergence-start velocity and doubles the divergence-limit velocity Further support additions yield only incremental increases in the divergence velocities.

The plate deflections are caused by slight misalignment in the flow. The flow is reaching the flat plates at a small angle of attack and causing lift and a leading-edge moment on the plates. This moment is destabilizing, so the plates continue to diverge as the freestream velocity increases. The deflections observed in the 0-, 2-, and 4-support cases are static, but they are dynamic in the 1-support run. For this case, the trailing edges of the plates vibrate when 15.0 m/s is reached, and the magnitude of the vibrations increases as the freestream velocity increases. These vibrations are a result of the vortices shed from the trailing edges of the flat plates. The Reynolds number based on plate thickness at 15.0 m/s for this run is approximately $1x10^3$, which is supercritical for shedding vortices. For the cases in which the deflections are static, the largest deflections always occur at the plate centers. Deflections are also noticeable at the leading and trailing edges between the structural supports and the test-section walls. These static leading- and trailing-edge deflections decrease in magnitude as the number of supports is increased.

The characteristic divergence velocities with the turbulence grid installed are presented in Table 4.2.

Number of Supports	Divergence-Start Velocity [m/s]	Divergence-Limit Velocity [m/s]
0	4.3	11.6
1	16.0	18.4
2	16.0	19.8
4	15.6	20.9
6	15.1	22.0

Table 4.2. Characteristic divergence velocities for runs with the turbulence grid installed.

These data are presented graphically in Figure 4.10. The results observed with the turbulence grid installed are about the same as for those without the grid. Again, the 1-support case shows dynamic deflections, while all other cases show static deflections.

For comparison, the experimental data are plotted in Figures 4.11 and 4.12 to show the effects of the turbulence grid on the divergence-start and divergence-limit velocities, respectively. As can be seen, the addition of the turbulence grid does not yield any considerable increases in the characteristic divergence velocities. In fact, a majority of the divergence velocities decrease with the turbulence grid installed, which can be attributed to the pressure loss across the grid.

To assess the flow uniformity across the test section, pressure drop measurements were taken across the LVFTDS at points between each plate. Figure 4.13 shows the pressure coefficient distribution across the LVFTDS for three freestream velocities without the turbulence grids. As seen, the pressure distribution is uniform for all three cases. In this figure, the pressure drop is given as

$$C_p = \frac{\Delta p}{q} = \frac{p_{\text{fore}} - p_{\text{aft}}}{\rho U^2/2}$$

The nominal value of $C_p \approx 0.4$ agrees with the computational predictions of Stage 1 for fully laminar flow.

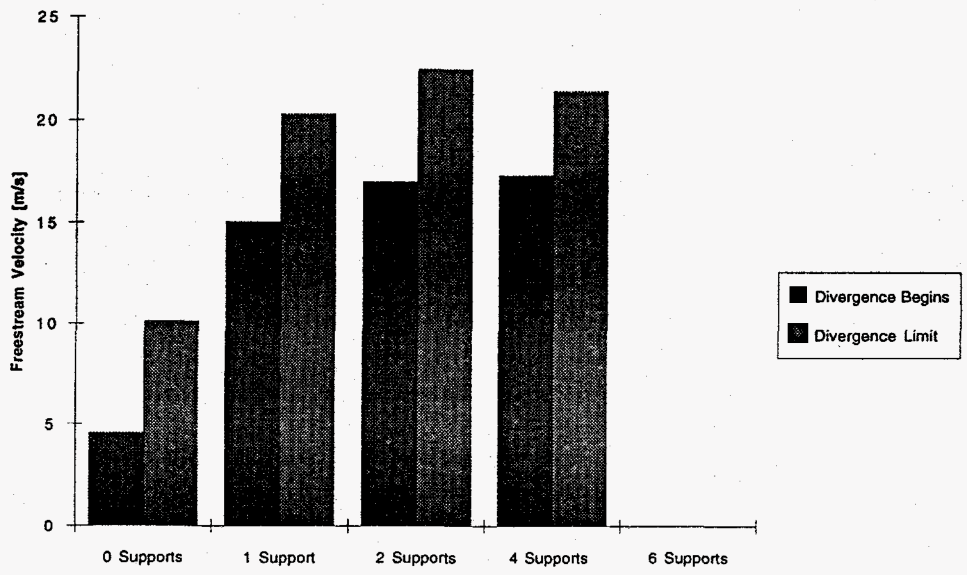

Figure 4.9. Characteristic divergence velocities without turbulence grid.

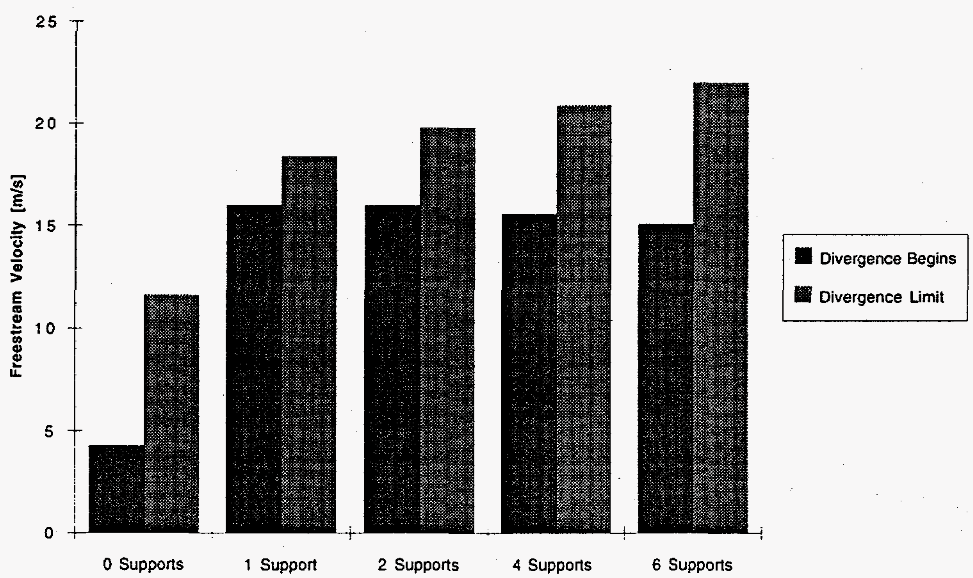

Figure 4.10. Characteristic divergence velocities with turbulence grid installed.

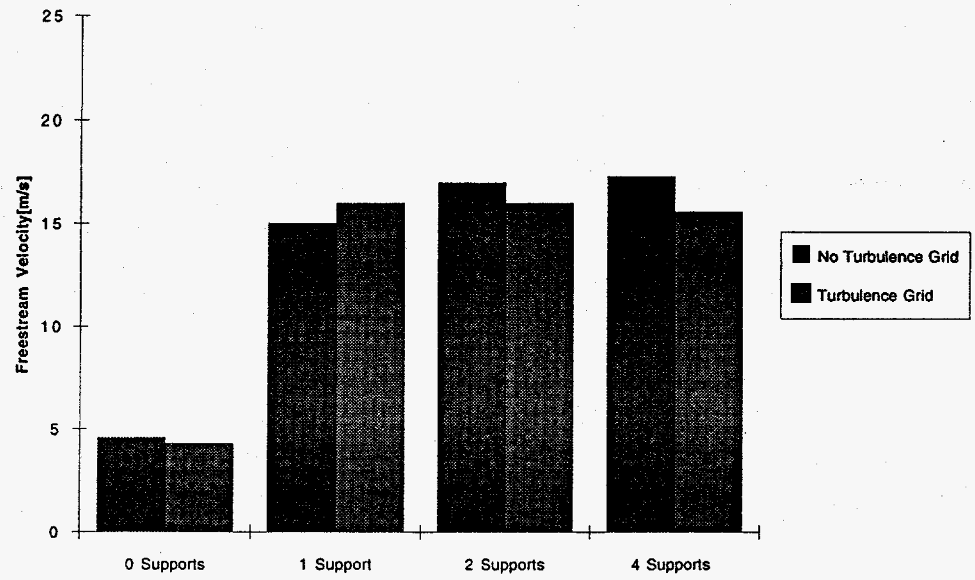

Figure 4.11. Divergence-start velocities with turbulence grid installed.

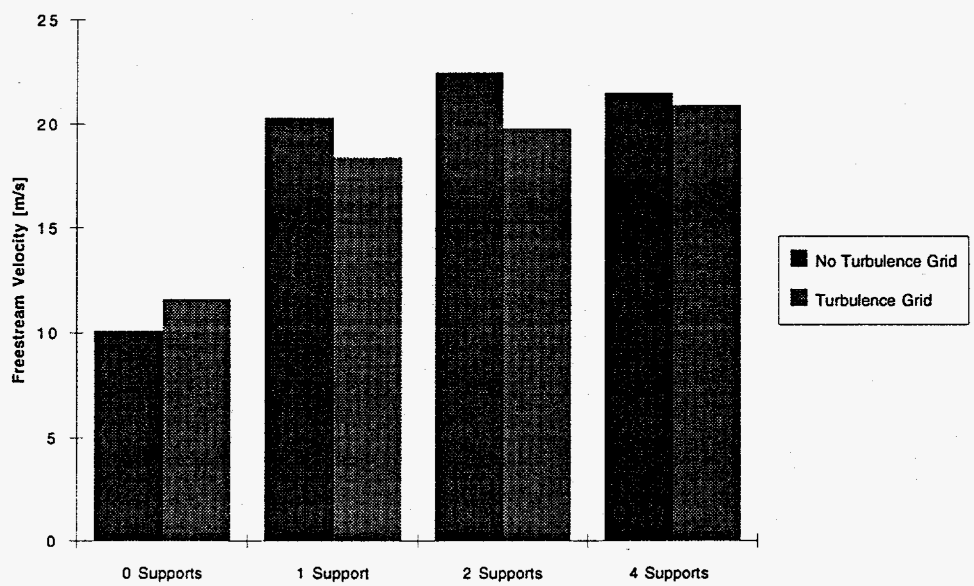

Figure 4.12. Divergence-limit velocities with turbulence grid installed.

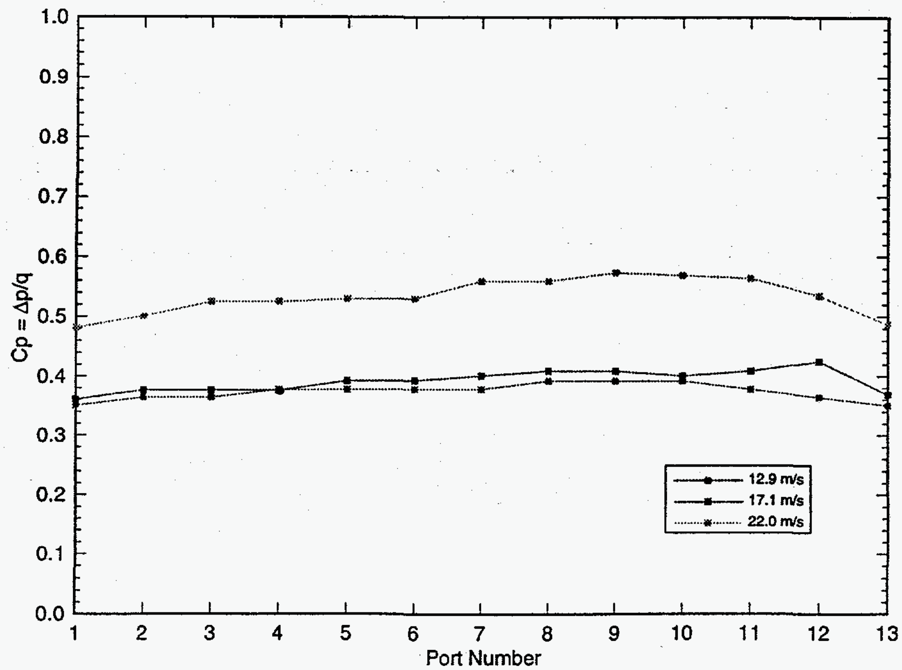

Figure 4.13. Pressure coefficient distribution across the test section.

5 CONCLUSIONS

A code has been developed for calculating the pressure drop across the LVFTDS.

A surrogate material has been found and used in the wind-tunnel tests in place of the actual detector material.

The work on this contract provided data on the divergence of the detector panels and offered a simple engineering solution to the problem. It is recommended that at least one set of fore and aft struts be used to support the panels.

The pressure drop data shown in Figure 4.13 provide a reasonable estimate of the expected pressure drop during the actual operation of the LVFTDS. In fact, the fully laminar results of the calculated C_p based on the code developed in Stage 1 agrees exactly with the experimental results. Thus this code can be used for other conditions.

Besides the data presented in this report, a video tape of the flow visualization was presented to the Contract Monitor, Dr. Russell Gritzo of LNL. It is demonstrated that the much of the basic design information can be obtained in this manner.

APPENDIX A

COMPUTED SHEAR STRESS AND PRESSURE DROP

Governing Equations

Inlet Reynolds Number

$$Re = \rho V h / \mu$$

Local Reynolds Number

$$Re(x) = \rho V x / \mu$$

Boundary Layer Thickness for Laminar Flow

$$\delta^{*} = 1.7208 / \ (Re(x))$$

Boundary Layer Thickness for Turbulent Flow

$$\delta^{*} = 1/8\ (0.16x/Re\verb|^|(1/7))$$

Centerline Local Velocity

$$U_2 = U_1\ [h / (h - 2\delta^{*})$$

Cross Sectional Flow Equation

$$\iint_{cs} \rho\mu\ \vec{v} \bullet \vec{n}\ \ dA$$

Wall Shear for Laminar Flow

$$\tau_{wall} = 0.332\ \rho^{(1/2)} \mu^{(1/2)} U^{1.5} / x^{(1/2)}$$

Wall Shear Turbulent Flow

$$\tau_{wall}\ \ 0.013\ \rho^{(6/7)} \mu^{(1/7)} U^{13/7} / x^{(1/7)}$$

Governing Equations

Wall Shear for the System

$$\int_0^{l_t} \tau_{wall,\ lamin}(x)\,dx + \int_{l_t}^{l} \tau_{wall,\ turb}(x)\,dx$$

Pressure Changes

$$-\rho\mu^2 h + \rho\int_0^h \mu^2(y)\,dy = (p_1 - p_2)h - 2\int_0^l \tau_{wall}\,dx$$

$$\Delta p = -\rho\mu^2 + \rho/h\int_0^h \mu^2(y)\,dy + 2/h\int_0^l \tau_{wall}\,dx$$

```
#include <stdio.h>
#include <math.h>
main()

{
  float lenth1, lenth2;
  float temp1, temp2, temp3, temp4, temp5, temp6;
  float presr1, presr2, presr3, presr4;
  float vel1, vel2;
  float spc1, spc;
  float densty1,densty3;
  float visc1, visc3;
  float dmin, vimax, re_min;
  double vmin, vmax;
  float dmax, vimin, re_max;
  float alpha_max;
  int num_real, num_alpha;

    fprintf(stderr,"INPUT: ");       /*Input for all variables*/

    fprintf(stderr,"\n\nDetector cell length along x direction of fluid flow (ft): ");
    scanf("%f", &lenth1);
    lenth2 = lenth1 / 3.281;
    fprintf(stderr,"\nDetector cell length along x direction of fluid flow (m):%f",lent

    fprintf(stderr,"\n\nTemp. Range Interval (Deg. C): "); /*Input for temp.*/
    scanf("%f", &temp1);
    fprintf(stderr,"-");
    scanf("%f",&temp2);
    temp3 = 273.15 + temp1;
    temp4 = 273.15 + temp2;
    fprintf(stderr,"\nTemp. Range Interval (Deg. K):%f - %f",temp3,temp4);
    temp5 = temp3 * 1.8;
    temp6 = temp4 * 1.8;
    fprintf(stderr,"\nTemp. Range Interval (Deg. R):%f - %f",temp5,temp6);

    fprintf(stderr,"\n\nPressure Range Interval (atm): "); /*Input for pressure*/
    scanf("%f", &presr1);
    fprintf(stderr,"-");
    scanf("%f", &presr2);
    presr3 = presr1 * 2116;
    presr4 = presr2 * 2116;
    fprintf(stderr,"\nPressure Range Interval (lbf/ft^2):%f - %f",presr3,presr4);

    fprintf(stderr,"\n\nInitial Fluid Flow Velocity Range Interval (m/s): "); /*Input f
    scanf("%f", &vel1);
    fprintf(stderr,"-");
    scanf("%f", &vel2);

    fprintf(stderr,"\n\nPlate space width (cm): "); /*Input for width*/
    scanf("%f", &spc);
    spc1 = spc / 100.0;
    fprintf(stderr,"\nPlate space width (m):%f", spc1);

   /*Will compute fluid density using the Ideal Gas Law*/

      densty1 = (presr3 / (1717 * temp5)) / 0.0019403;
      densty3 = (presr4 / (1717 * temp6)) / 0.0019403;


   /*Will compute fluid viscosity using Sutherland Formula*/

      visc1 = pow((temp3 / 273.111111),(3.0/2)) * ((273.111111 + 110.555556) /(temp3 +
      visc3 = pow((temp4 / 273.111111),(3.0/2)) * ((273.111111 + 110.555556) /(temp4 +

   /*Will compute a min, midpoint, & max entrance reynolds number corresponding to the
```

```c
    if(densty1 < densty3)
         dmin = densty1;
    else
         dmin = densty3;

    if(visc1 > visc3)
         vimax = visc1;
    else
         vimax = visc3;

    vmax = vimax / dmin;
    re_min = spc1 * vel1 / vmax;

    fprintf(stderr,"vmax:%f",vmax);



    if(densty1 > densty3)
         dmax = densty1;
    else
         dmax = densty3;

    if(visc1 < visc3)
         vimin = visc1;
    else
         vimin = visc3;

    vmin = vimin / dmax;
    re_max = spc1 * vel2 / vmin;

    fprintf(stderr,"vmin:%f",vmin);
    fprintf(stderr,"\n\nMin. Viscosity Value ((N*s)/m^2):%f",vimin);
    fprintf(stderr,"\nMax. Viscosity Value ((N*S)/m^2):%f",vimax);

    fprintf(stderr,"\n\nMin. Kinematic Viscosity Value (m^2/s):%f",vmin);
    fprintf(stderr,"\nMax. Kinematic Viscosity Value (m^2/s):%f",vmax);

    fprintf(stderr,"\n\nMin. Density Value (kg/m^3):%f",dmin);
    fprintf(stderr,"\nMax. Density Value (kg/m^3):%f",dmax);

    fprintf(stderr,"\n\nMin. Entrance Re Number:%f",re_min);
    fprintf(stderr,"\nMax. Entrance Re Number:%f",re_max);

    alpha_max = lenth2 / spc1;
    fprintf(stderr,"\n\nMaximum alpha is:%f",alpha_max);

    fprintf(stderr,"\n\nNumber of desired Re numbers: ");
    scanf("%d", &num_real);

    fprintf(stderr,"\n\nNumber of desired alpha's: ");
    scanf("%d", &num_alpha);

    printf("\n\n%f %f %d %f %d",re_min,re_max,num_real,alpha_max,num_alpha);
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define TOL 1
#define TRANS 250000

main(int argc, char **argv)
{

int re_index, num_re;
int alpha_indx, num_alpha, alpha_ind;
float re_in, re_max, re_min;
float alpha, alpha_max, alpha_min;
float re_x, beta, gama, re_temp, err;
int scan_err = 0;

if(argc == 6)
{
  re_min = atof(argv[1]);
  re_max = atof(argv[2]);
  num_re = atoi(argv[3]);
  alpha_max = atof(argv[4]);
  num_alpha = atoi(argv[5]);
}if(argc == 1){
                scan_err += scanf("%e",&re_min);
                scan_err += scanf("%e",&re_max);
                scan_err += scanf("%d",&num_re);
                scan_err += scanf("%e",&alpha_max);
                scan_err += scanf("%d",&num_alpha);
                }
else {
     fprintf(stderr,"wrong # of args.");
     exit(1);
     }

fprintf(stderr, "re_min = %f\n", re_min);
fprintf(stderr, "re_max = %f\n", re_max);
fprintf(stderr, "num_re = %d\n", num_re);
fprintf(stderr, "alpha_max = %f\n", alpha_max);
fprintf(stderr, "num_alpha = %d\n", num_alpha);


printf("    Variables = \"re_x\", \"alpha\", \"beta\", \"gama\"\n");

for(re_index = 0; re_index <= num_re; re_index++)
    {
     re_in = (((re_max-re_min)/num_re) * re_index) + re_min;

     printf("\n\n    #Inlet Reynolds Number:%f",re_in);
     printf("\n\n    #Local Re #       Alpha       Beta        Gama\n");
     printf("\n    zone i = %d, f = point, t = \"re_in = %lf\"\n\n",num_alpha, re_in);

     for(alpha_indx = 1; alpha_indx <= num_alpha; alpha_indx++)
         {
          alpha = alpha_max / num_alpha * alpha_indx;
          re_x = alpha * re_in;

             do
               {
                if(re_x < TRANS)        /*laminar*/
                   beta = 1.7208 * alpha / (powf(re_x,(1.0/2)));
                else                    /*turbulent*/
                   beta = 0.125 * 0.16 * alpha / (powf(re_x,(1.0/7)));

                gama = 1.0 / (1.0 - 2.0 * beta);
```

```
                re_temp = re_in * alpha * gama;
                err = fabs(re_x - re_temp);
                re_x = re_temp;
              }
              while(err > TOL);
              printf("\n    %-11f  %-10f  %-10f  %-10f",re_x,alpha,beta,gama);
        }
    }
}
```

```c
#include <stdio.h>
#include <math.h>
main()

{
  float lenth1, lenth2, width, width1;
  float temp1, temp3, temp5;
  float presr1, presr3;
  float vel1;
  float spc1, spc;
  float densty1;
  float visc1, re_in;
  double k_visc;
  int num_intvl, numy_intvl;

    fprintf(stderr,"\n\nINPUT: ");      /*Input for all variables*/

    fprintf(stderr,"\n\nDetector cell length along x direction of fluid flow (ft): ");
    scanf("%f", &lenth1);
    lenth2 = lenth1 / 3.281;
    fprintf(stderr,"\nDetector cell length along x direction of fluid flow (m):%f",lent

    fprintf(stderr,"\n\nDetector cell width along z direction of coord. system (ft): ")
    scanf("%f", &width1);
    width = width1 / 3.281;
    fprintf(stderr,"\nDetector cell width along z direction of coord. system (m):%f",wi


    fprintf(stderr,"\n\nTemperature (Deg. C): "); /*Input for temp.*/
    scanf("%f", &temp1);
    temp3 = 273.15 + temp1;
    fprintf(stderr,"\nTemperature (Deg. K):%f",temp3);
    temp5 = temp3 * 1.8;
    fprintf(stderr,"\nTemperature (Deg. R):%f",temp5);

    fprintf(stderr,"\n\nPressure (atm): "); /*Input for pressure*/
    scanf("%f", &presr1);
    presr3 = presr1 * 2116;
    fprintf(stderr,"\nPressure (lbf/ft^2):%f",presr3);

    fprintf(stderr,"\n\nInitial Fluid Flow Velocity (m/s): "); /*Input for Vel.*/
    scanf("%f", &vel1);

    fprintf(stderr,"\n\nPlate space width (cm): "); /*Input for width*/
    scanf("%f", &spc);
    spc1 = spc / 100.0;

    fprintf(stderr,"\n\nNumber of intervals along x: ");
    scanf("%d", &num_intvl);

    fprintf(stderr,"\n\nNumber of intervals along y: ");
    scanf("%d", &numy_intvl);

    /*Will compute fluid density using the Ideal Gas Law*/

       densty1 = (presr3 / (1717 * temp5)) / 0.0019403;

    /*Will compute fluid viscosity using Sutherland Formula*/

       visc1 = pow((temp3 / 273.111111),(3.0/2)) * ((273.111111 + 110.555556) /(temp3 +

    /*Will compute an entrance reynolds number*/

       k_visc = visc1 / densty1;
       re_in = spc1 * vel1 / k_visc;
```

```
fprintf(stderr,"\n\nspc1:%e",spc1);
fprintf(stderr,"\n\nvel1:%e",vel1);
fprintf(stderr,"\n\nk_visc:%e",k_visc);

fprintf(stderr,"\n\nViscosity ((N*s)/m^2):%e",visc1);

fprintf(stderr,"\n\nKinematic Viscosity (m^2/s):%lf",k_visc);

fprintf(stderr,"\n\nDensity (kg/m^3):%f",densty1);

fprintf(stderr,"\n\nEntrance Re Number:%f\n\n\n\n",re_in);

printf("%e %d %e %e %e %e %e %e %d %e\n",lenth2,num_intvl,re_in,spc1,visc1,k_vis
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define TOL 1
#define TRANS 250000

main(int argc, char **argv)
{

float lenth, heit, dist_min, re_in, re_x;
float viscl, k_visc, densty1, init_vel;
float alpha_max, alpha, dist_x;
float delta_str, locl_vel, re_temp, err;
float twall, twall1, twall2, twall3, twall4, twall5;
int num_intvl, num_alpha, alpha_indx;
int scan_err=0;
float presr1, presr2, presr_tot;
float dist_y, vel_y, vel_y1, vel_y2, vel_y3, vel_y4;
float spc_invl, width, force1, force2, force3, delta;
int numy_intvl, heit_indx;


if(argc == 11)
{
  lenth = atof(argv[1]);
  num_intvl = atoi(argv[2]);
  re_in = atof(argv[3]);
  heit = atof(argv[4]);
  viscl = atof(argv[5]);
  k_visc = atof(argv[6]);
  densty1 = atof(argv[7]);
  init_vel = atof(argv[8]);
  numy_intvl = atoi(argv[9]);
  width = atof(argv[10]);
}if(argc ==1) {
        scan_err += scanf("%e",&lenth);
        scan_err += scanf("%d",&num_intvl);
        scan_err += scanf("%e",&re_in);
        scan_err += scanf("%e",&heit);
        scan_err += scanf("%e",&viscl);
        scan_err += scanf("%e",&k_visc);
        scan_err += scanf("%e",&densty1);
        scan_err += scanf("%e",&init_vel);
        scan_err += scanf("%d",&numy_intvl);
        scan_err += scanf("%e",&width);}
else{
        fprintf(stderr, "wrong # of args.");
        exit(1);
}

fprintf(stderr, "lenth = %f\n", lenth);
fprintf(stderr, "num_intvl = %d\n", num_intvl);
fprintf(stderr, "re_in = %f\n", re_in);
fprintf(stderr, "heit = %f\n", heit);
fprintf(stderr, "viscl = %f\n", viscl);
fprintf(stderr, "k_visc = %f\n", k_visc);
fprintf(stderr, "densty = %f\n", densty1);
fprintf(stderr, "init_vel = %f\n", init_vel);
fprintf(stderr, "numy_intvl = %d\n", numy_intvl);
fprintf(stderr, "width = %f\n", width);

dist_min = lenth/num_intvl;
twall5 = ((lenth-dist_min)/num_intvl)/2.0;

spc_invl = heit/numy_intvl;
vel_y3 = ((heit-spc_invl)/numy_intvl)/2.0;
```

```
printf("    Variables = \"re_x\", \"dist_x\", \"locl_vel\", \"delta_str\"\n");

       printf("\n\n    #Inlet Reynolds Number:%f",re_in);
       printf("\n\n    #Local Re #        Length(m)   Locl_vel(m/s)    delta_str(m)\n");
       printf("\n    zone i = %d, f = point, t = \"re_in = %f\"\n\n",num_intvl, re_in);
       alpha_max = lenth / heit;
       num_alpha = num_intvl;

       for(alpha_indx = 1; alpha_indx <= num_alpha; alpha_indx++)
           {
            alpha = alpha_max / num_alpha * alpha_indx;
            re_x = alpha * re_in;
            dist_x = alpha * heit;

               do
                 {
                  if(re_x < TRANS)        /*laminar*/
                     delta_str = 1.7208 * dist_x / (powf(re_x,(1.0/2)));
                  else                   /*turbulent*/
                     delta_str = 0.125 * 0.16 * dist_x / (powf(re_x,(1.0/7)));

                  locl_vel = init_vel * heit / (heit - 2 * delta_str);

                  re_temp = locl_vel * dist_x / k_visc;
                  err = fabs(re_x - re_temp);
                  re_x = re_temp;
                 }
                while(err > TOL);
                printf("\n    %-11f  %-10f  %-10f  %-10f",re_x,dist_x,locl_vel,delta_s
           if(re_x < TRANS)
              twall = 0.365 * (powf(densty1,(0.5))) * (powf(visc1,(0.5))) * (powf(locl_
           else
              twall = 0.0135 * (powf(densty1,(6.0/7))) * (powf(visc1,(1.0/7))) * (powf(

           if(alpha_indx == 1 | alpha_indx == num_alpha)
             twall1 = twall;
           else
             twall1 = twall * 2.0;

           if(alpha_indx == 1)
             twall3 = 0.0;
           else
             twall3 = twall2;

           twall2 = twall1 + twall3;

           }
            printf("\n");
            twall4 = twall5 * twall2;
            printf("\nShear Stress (kg/m*s^2):%f\n\n",twall4);


            presr1 = -1.0 * densty1 * pow(init_vel,2.0);
            printf("\n\nInlet pressure (kg/m*s^2):%f\n\n",presr1);

         for(heit_indx = 1; heit_indx <= numy_intvl; heit_indx++)
           {

           dist_y = (heit_indx * spc_invl);

           if(re_x > TRANS)
             {
             delta = .16 * dist_x / (pow(re_x,(1.0/7)));
             vel_y = locl_vel * (pow((dist_y / delta),(1.0/7)));
```

```
        }
      else
        {
        delta = 5.5 * dist_x / (pow(re_x,(1.0/2)));
        vel_y = locl_vel * ((2 * dist_y / delta) - (pow(dist_y,2.0)/pow(delta,2.0)
        }
      if(heit_indx == 1)
        vel_y1 = 0.0;
      else
        vel_y1 = vel_y2;

      vel_y2 = vel_y1 + vel_y;
      }
      vel_y4 = vel_y2 * vel_y3;
      presr2 = densty1 * pow(vel_y4,2.0);
      printf("\n\nPressure at length-x (kg/m*s^2):%f",presr2);


      presr_tot = presr1 + presr2 + 2.0 * twall4 * -1.0;
      printf("\n\nTotal pressure change within the system (kg/m*s^2):%f",presr_tot

      force1 = presr1 * heit * width;
      printf("\n\n\nForce at the inlet of the system (N):%f",force1);

      force2 = presr2 * heit * width;
      printf("\n\nForce at the length x of the system (N):%f",force2);

      force3 = twall4 * width * lenth * -1.0;
      printf("\n\nShear force (N):%f\n\n",force3);
}
```

One Foot by One Foot Detecting Cell

Viscosity - (N*s)/m^2 = 1.837e-5
Kinematic Viscosity - m^2/s = 1.600e-5
Pressure = 1 atm = 2.06e-7 lb/ft^2
Density - kg/m^3 = 1.184 = 2.8641 slugs/ft^3

Inlet Re#	Outlet Re#	Initial Vel. (m/s)	Final Vel. (m/s)
6441.1768	116023.5078	5	5.9099
12882.3535	220971.875	10	11.2558
19323.5293	324819.5313	15	16.5456
25764.707	434054.5938	20	22.1098
32205.8848	540804	25	27.5474
38647.0586	6477292.75	30	32.9717
45088.2383	753571.125	35	38.3853
51529.4141	811620.375	40	42.7899
57990.5938	965626.8125	45	49.1869
64411.7695	1071448.625	50	54.5773
70852.9453	1177154.75	55	59.6918

Shear Force (N)	Initial Vel. (ft/s)	Final Vel. (ft/s)	Inlet Pressure (Pa)	FINAL Cp
-2.01E-03	16.404	19.3891999	0.149545744	0.4
-5.44E-03	32.808	36.9280286	0.299091483	
-0.01271	49.212	54.2828045	0.448637222	
-0.02488	65.616	72.5378318	0.598182962	
-0.04017	82.02	90.3775099	0.747728701	
-0.05904	98.424	108.173553	0.89727444	
-0.08069	114.828	125.934492	1.04682018	
-0.1065	131.232	140.385104	1.196365919	
-0.1346	147.636	161.372382	1.345911658	
-0.1665	164.04	179.057206	1.495457398	
-0.2007	180.444	196.722673	1.645003137	