Dynamic Pressure and Airspeed Measurements in a Low-Speed Wind Tunnel

Parham Khodadi*
A E 303, Section 3, with Dr. Xiaofeng Liu

This experiment investigates the uniformity of the flow field in a low-speed wind tunnel using a total pressure rake. The study examines the configuration and operating principles of the wind tunnel while applying Bernoulli's equation to determine dynamic pressure and velocity measurements in free-stream conditions. Additionally, a contour map is utilized to visualize the two-dimensional streamwise velocity distribution across the test section.

I. Nomenclature

q	=	Dynamic pressure (Pa)
$\bar{q}$	=	Mean dynamic pressure (Pa)
V	=	Airspeed (m/s)
$\bar{V}$	=	Mean airspeed (m/s)
ρ	=	Air density (kg/m^3)
IAS	=	Indicated Airspeed (m/s)
p_0	=	Total pressure (Pa)
p_s	=	Static pressure (Pa)
Re	=	Reynolds number
μ	=	Dynamic viscosity of air (Pa·s)
q_mean	=	Mean dynamic pressure (Pa)
V_∞	=	Freestream velocity (m/s)
T	=	Temperature (K or °C)
$P_{ambient}$	=	Ambient pressure (Pa or inHg)
ρ_∞	=	Freestream air density (kg/m^3)
Δp	=	Pressure differential (Pa)

II. Introduction

Flow uniformity in a wind tunnel test section is crucial for reliable aerodynamic testing. This experiment evaluates dynamic pressure and airspeed variations using a Pitot tube rake in the SDSU low-speed wind tunnel.

Total and static pressures were recorded at three test conditions (0, 2, and 5 inH2O), and airspeed was computed using Bernoulli's equation. A convergence test determined the number of samples required for stable time-averaged pressure measurements, with a threshold of 0.005%.

The results provide insight into test section uniformity, measurement stability, and wind tunnel performance.

III. Theory

A. Velocity Measurement Using Pitot Tubes

The uniformity of flow in a wind tunnel test section is essential for reliable aerodynamic testing. A Pitot tube rake is used to measure total pressure, while a separate static pressure probe (Port 7) provides a reference static pressure, assuming negligible lateral pressure gradients. Velocity is determined using Bernoulli's equation:

*Aerospace Engineering student, San Diego State University

$$V = \sqrt{\frac{2q}{\rho}} \tag{1}$$

where the dynamic pressure q is:

$$q = p_0 - p_s \tag{2}$$

where p_0 is total pressure, p_s is static pressure, and ρ is air density. The same calculation is applied at the test section inlet using a separate Pitot tube (Ports 61 & 62) to determine the Indicated Airspeed (IAS).

B. Flow Uniformity and Data Reduction

To assess flow uniformity, deviations from the mean dynamic pressure $\bar{q}$ are computed as:

$$\frac{q - \bar{q}}{\bar{q}} \times 100\% \tag{3}$$

Similarly, airspeed deviations are evaluated as:

$$\frac{V - \bar{V}}{\bar{V}} \times 100\% \tag{4}$$

These calculations quantify test section non-uniformities and identify variations in airflow.

C. Convergence Testing for Dynamic Pressure

A convergence test determines the minimum number of samples required for stable time-averaged dynamic pressure. The convergence percentage is calculated as:

$$\%\text{Convergence} = \left| \frac{x_{i+1} - x_i}{x_i} \right| \times 100 \tag{5}$$

where x_i is the cumulative mean dynamic pressure up to the ith sample. A threshold of 0.005% is used to confirm stability, ensuring accurate measurements.

IV. Experimental Setup

The experiment was conducted in a subsonic wind tunnel at San Diego State University, as depicted in Figure 1. The wind tunnel is designed to provide a uniform flow field for aerodynamic testing and is equipped with a total pressure rake to measure flow uniformity.

A Pitot tube rake with 35 measurement ports was used to capture total and static pressure values at various locations. The static pressure probe was located at port 7, and ports 61 and 62 recorded the static and total pressures at the test section inlet. A Fisherbrand digital thermometer (Figure 2) monitored ambient conditions, including temperature, humidity, and atmospheric pressure, to ensure accurate calculations.

A data acquisition system collected pressure readings at a sampling rate of 600 samples per test condition. The measured dynamic pressure was used to determine the indicated airspeed (IAS) at different pressure settings of 2 inH_2O and 5 inH_2O.

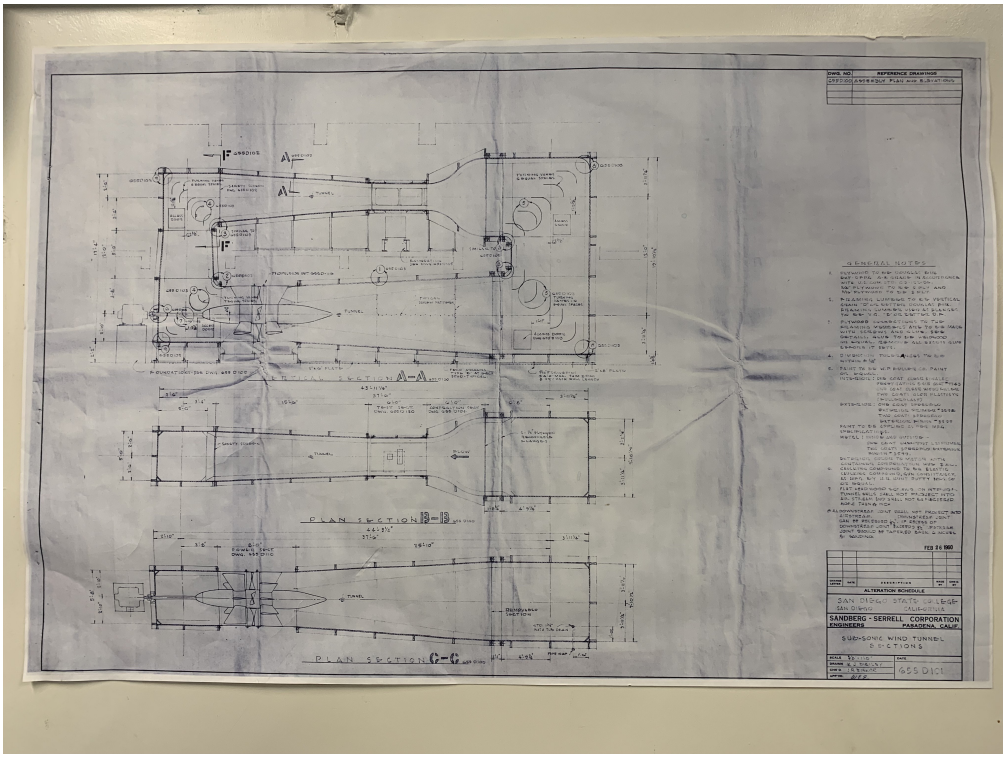

Fig. 1 Blueprint of the subsonic wind tunnel used in the experiment.

Fig. 2 Digital thermometer monitoring ambient conditions.

V. Procedures

A. Wind Tunnel Operation

1) The wind tunnel was set to the desired dynamic pressure condition (0, 2, and 5 inH_2O).
2) The fan speed was adjusted to ensure steady-state conditions before recording data.
3) The Pitot tube rake and static pressure probe were verified for proper alignment.

B. Data Collection

1) A digital pressure scanner recorded total and static pressure values at all 35 Pitot tube locations.
2) Pressure readings were logged at a sampling rate sufficient for statistical stability.
3) Each test condition was run for 600 samples to ensure sufficient data for convergence analysis.

C. Data Processing

1) Dynamic pressure at each port was computed as:

$$q = P_{total} - P_{static} \tag{6}$$

2) Airspeed was calculated using Bernoulli's equation.
3) Flow uniformity was analyzed using deviation plots.
4) A convergence test determined the number of samples required for stable time-averaged pressure.

VI. Results and Data Reduction

A. Measured Values

Table 1 Experimental Measurements Summary

Test Condition	Mean Dynamic Pressure (psi)	Indicated Airspeed (IAS) (m/s)	Unit Reynolds Number
0 inH_2O	0.000	0.000	N/A
2 inH_2O	0.072	28.526	4.748×10^4
5 inH_2O	0.181	45.103	7.464×10^4

B. Flow Uniformity Analysis

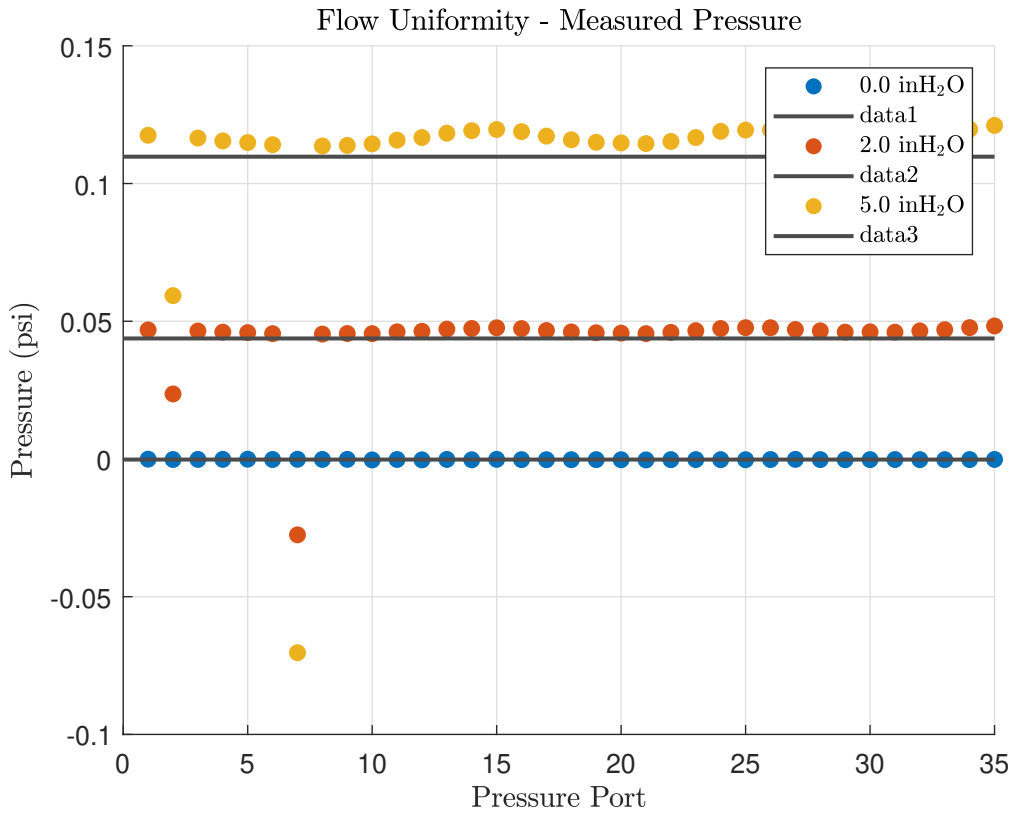

Fig. 3 Averaged pressure for three runs at dynamic pressures settings of 0.0inH_2O, 2.0inH_2O and 7.0 inH_2O. The black line is the average pressure for all pressure ports at a given dynamic pressure setting. The red rectangles mark points of interest.

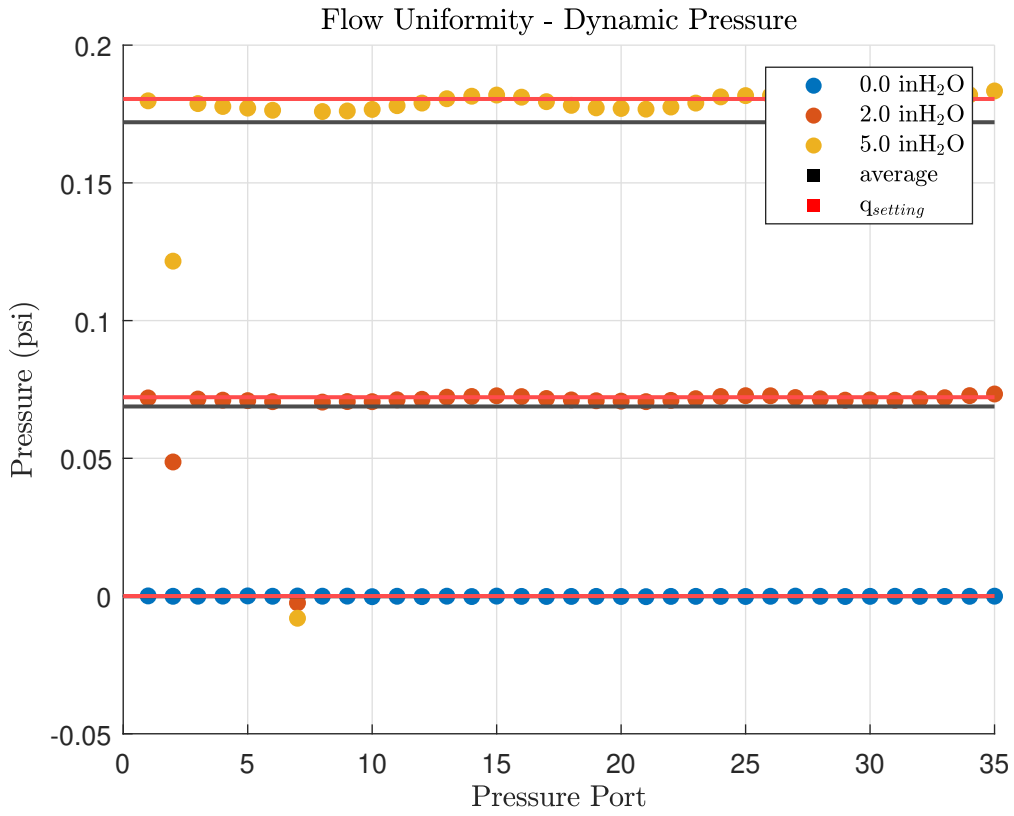

Fig. 4 Averaged measured dynamic pressure for three runs at indicated dynamic settings of 0.0inH_2O, 2.0inH_2O and 7.0 inH_2O. The black line is the average pressure for all pressure ports at a given dynamic pressure setting. The red rectangles mark points of interest.

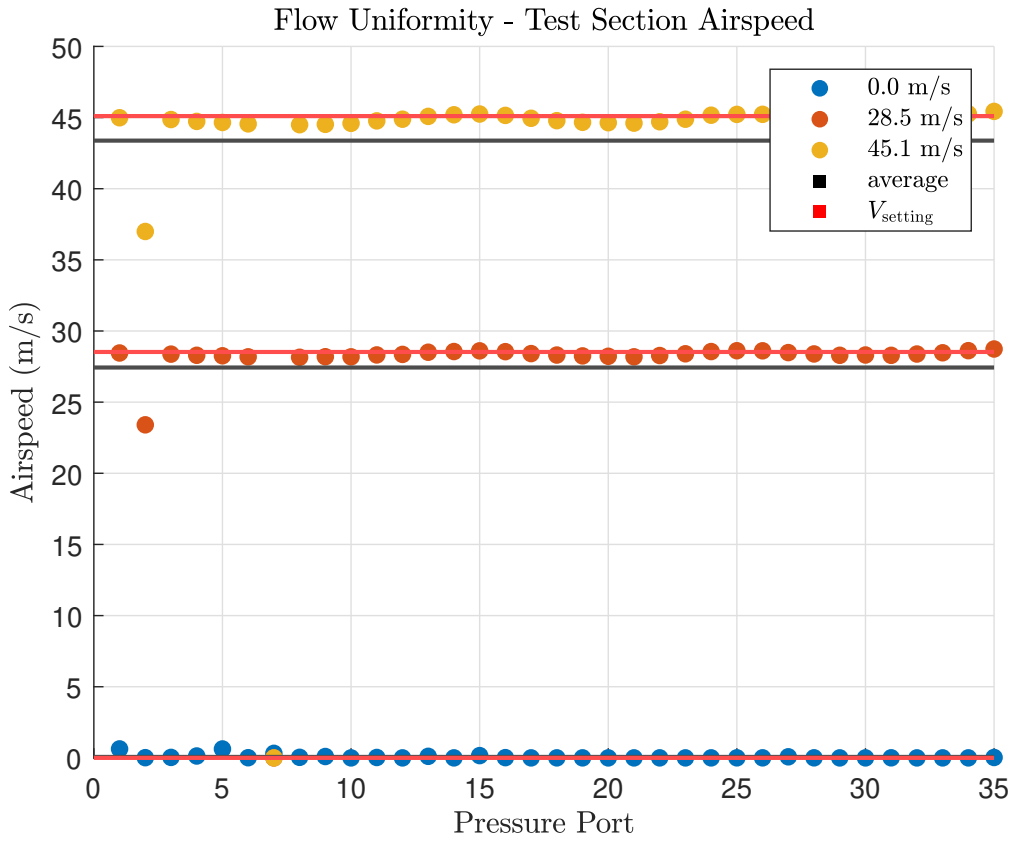

Fig. 5 Averaged measured airspeed for three runs at airspeeds settings of 0 m/s, 28.5 m/s, 45.1 m/s. The black line is the average airspeed for all pressure ports at a given airspeed setting. The red rectangles mark points of interest

C. Dynamic Pressure Uniformity

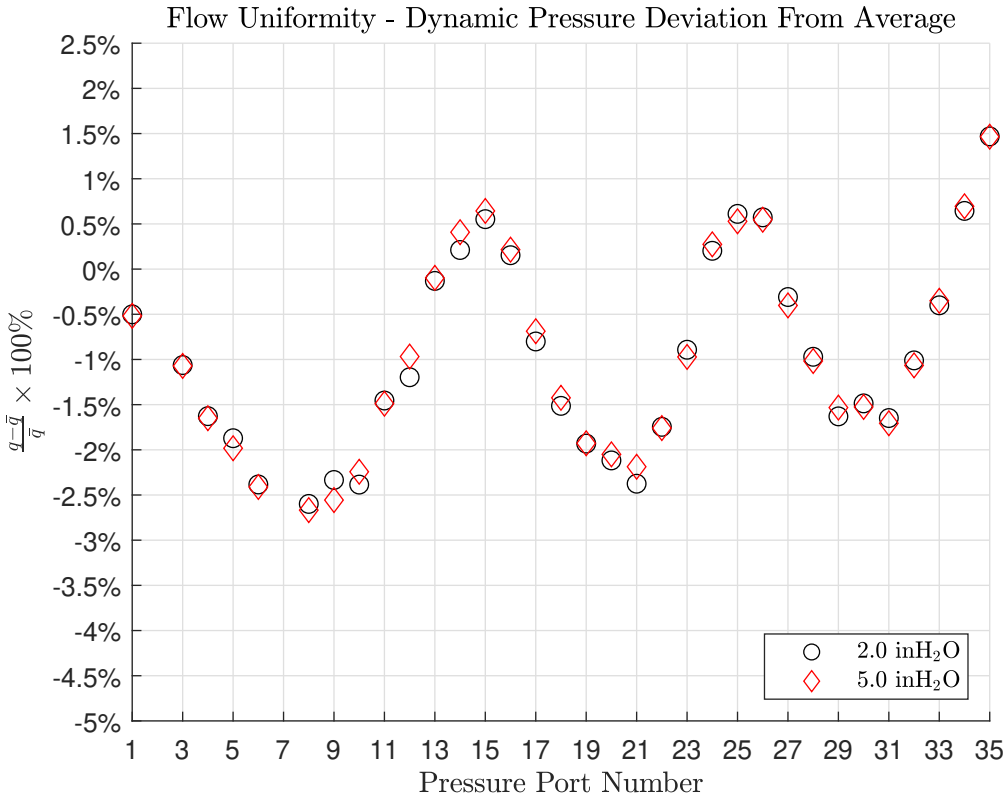

Fig. 6 Flow uniformity for dynamic pressure setting of 2 inH$_2$O and 5 inH$_2$O.

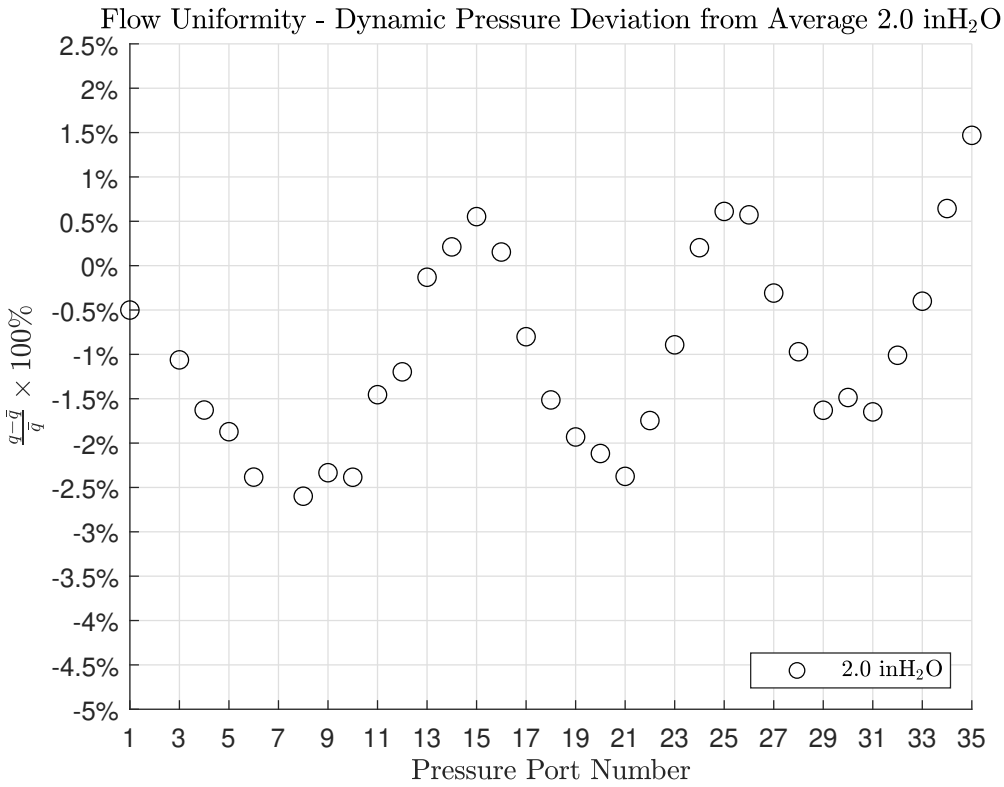

Fig. 7 Flow uniformity for dynamic pressure setting of 2 inH_2O.

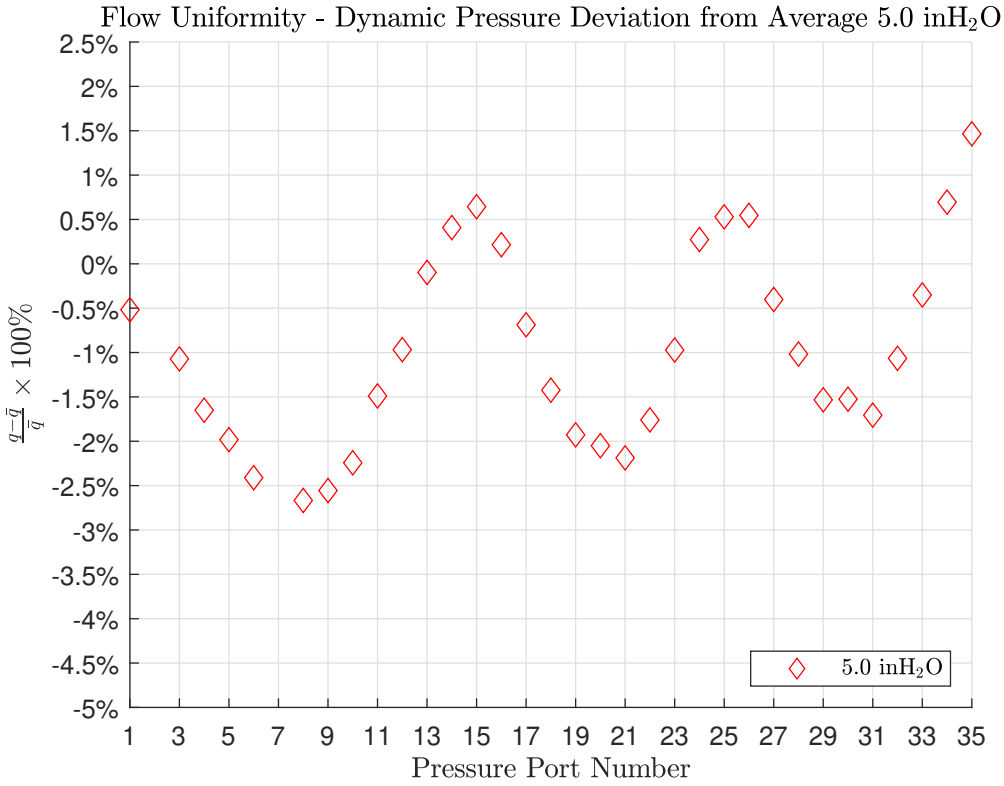

Fig. 8 Flow uniformity for dynamic pressure setting of 5 inH_2O.

D. Airspeed Uniformity

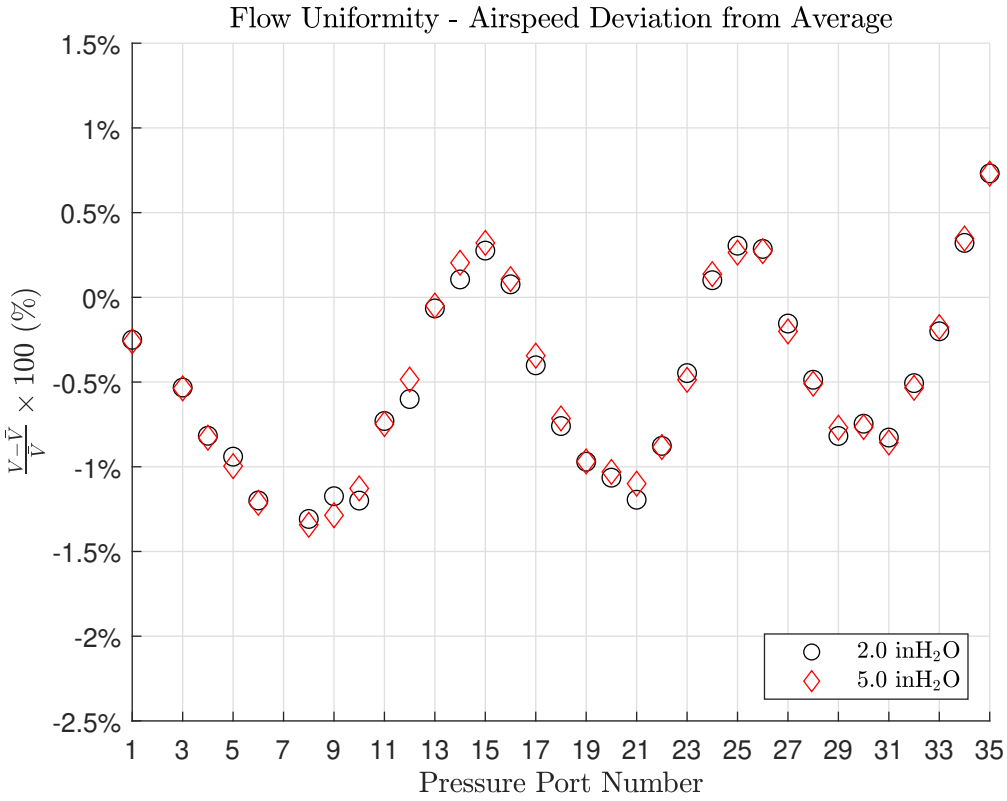

Fig. 9 Airspeed uniformity for dynamic pressure settings of 2 inH$_2$O and 5 inH$_2$O.

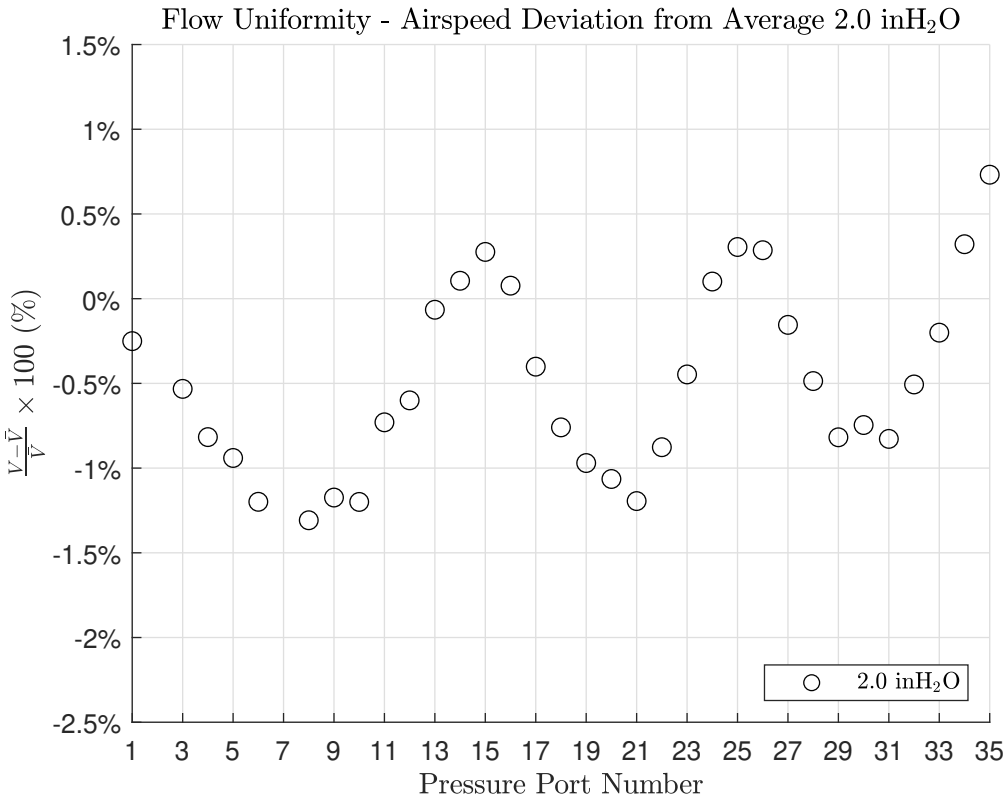

Fig. 10 Airspeed uniformity for dynamic pressure settings of 2 inH_2O.

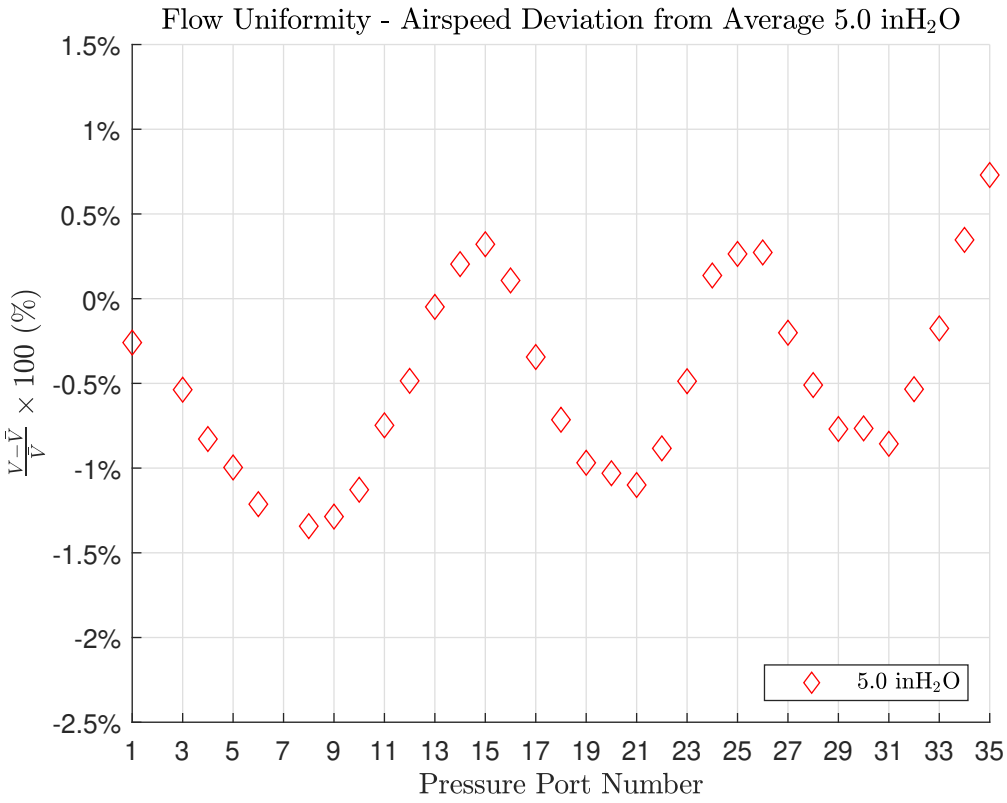

Fig. 11 Airspeed uniformity for dynamic pressure settings of 5 inH_2O.

E. Dynamic Pressure Contours

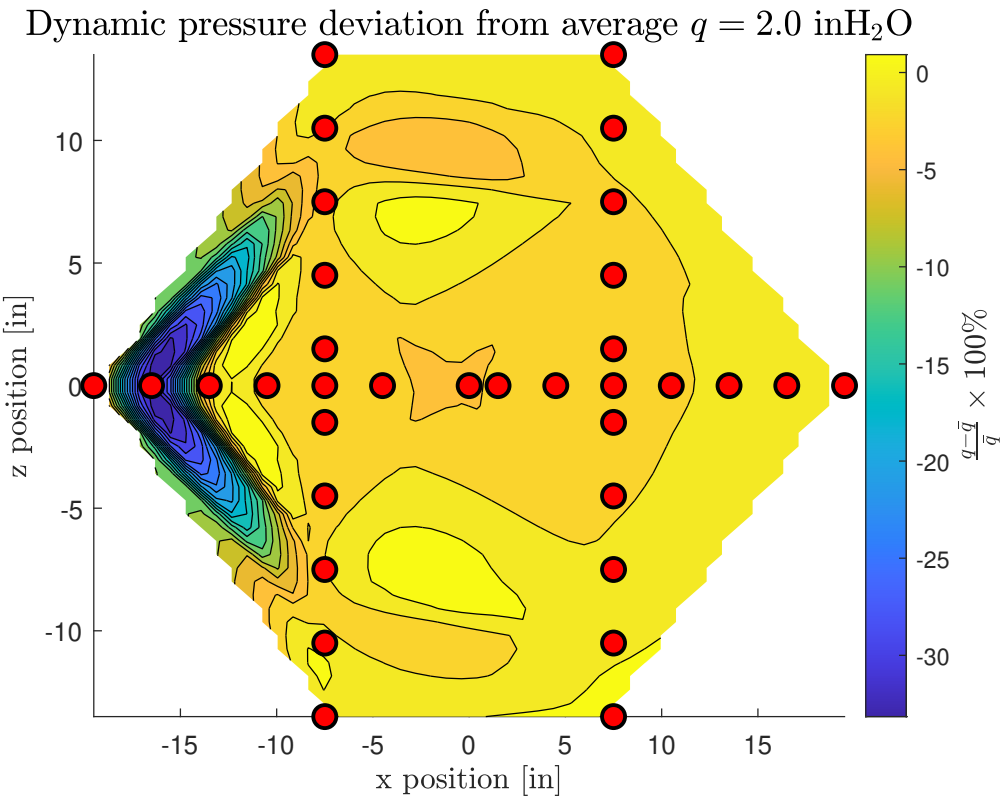

Fig. 12 Dynamic pressure fluctuation map for dynamic pressure setting of 2.0 inH$_2$O.

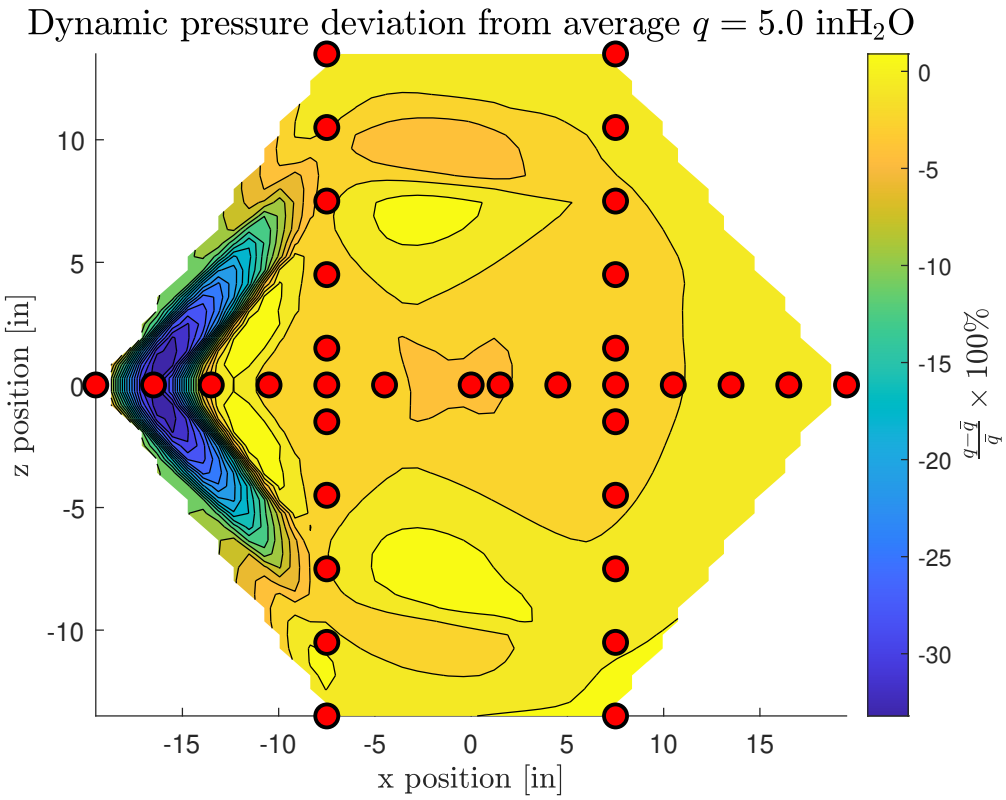

Fig. 13 Dynamic pressure fluctuation map for dynamic pressure setting of 5.0 inH$_2$O.

F. Airspeed Contours

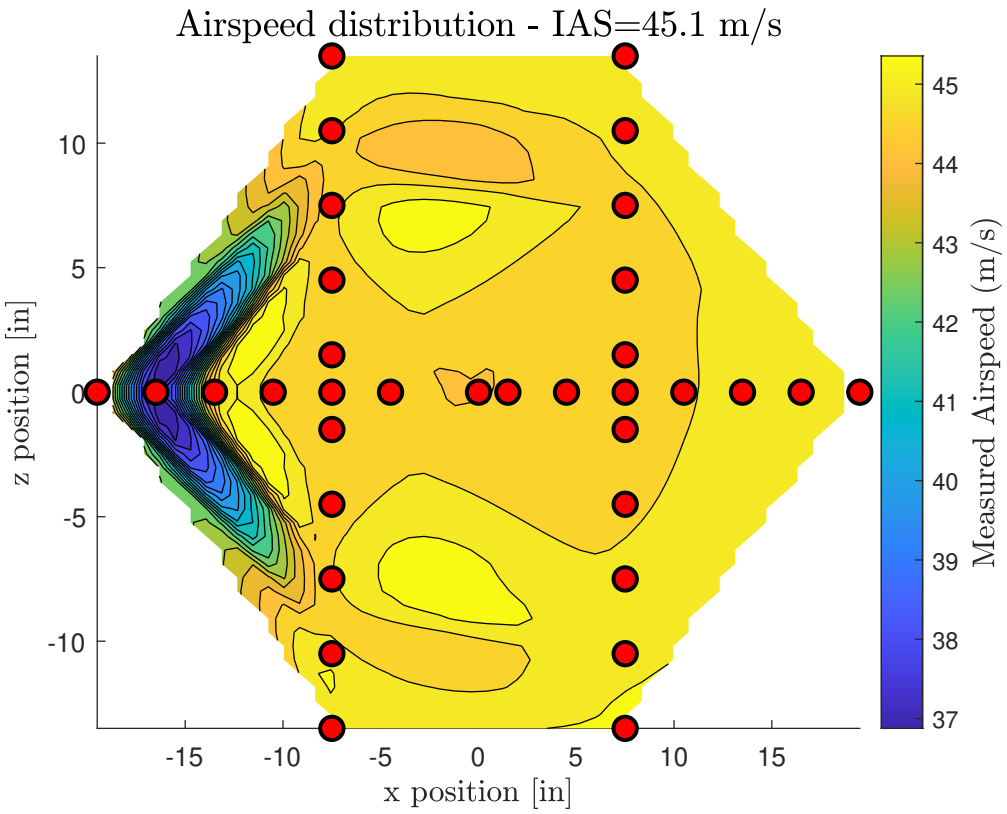

Fig. 14 Airspeed map for IAS 28.5 m/s.

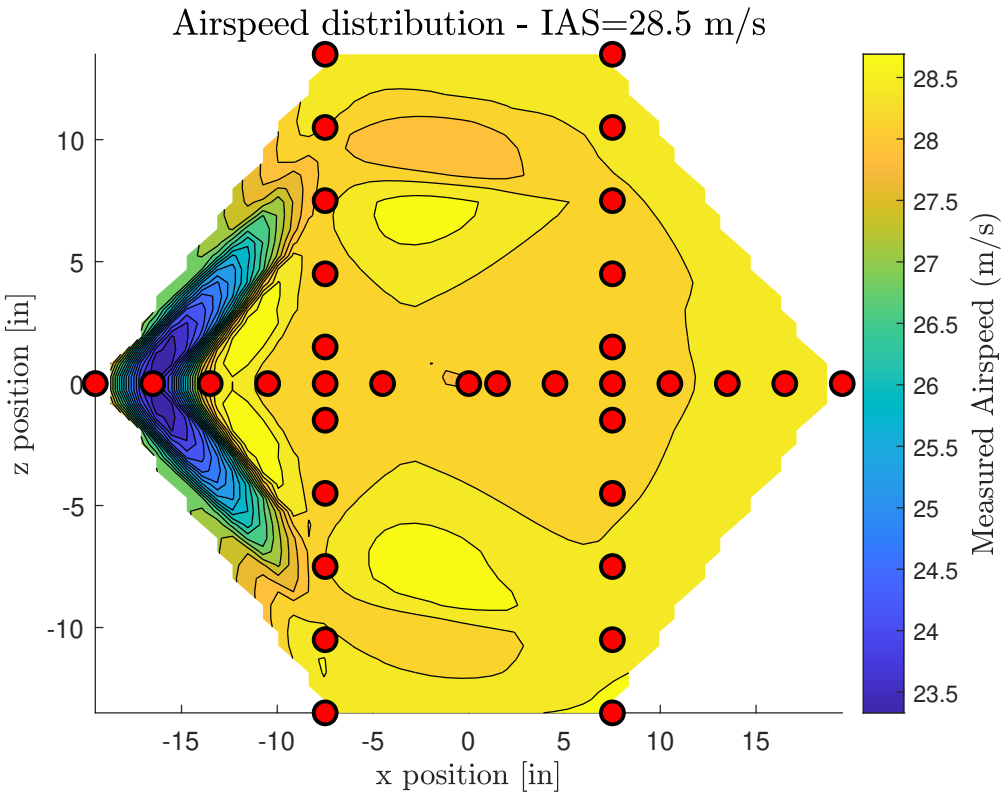

Fig. 15 Airspeed map for IAS 45.1 m/s.

G. Convergence Test

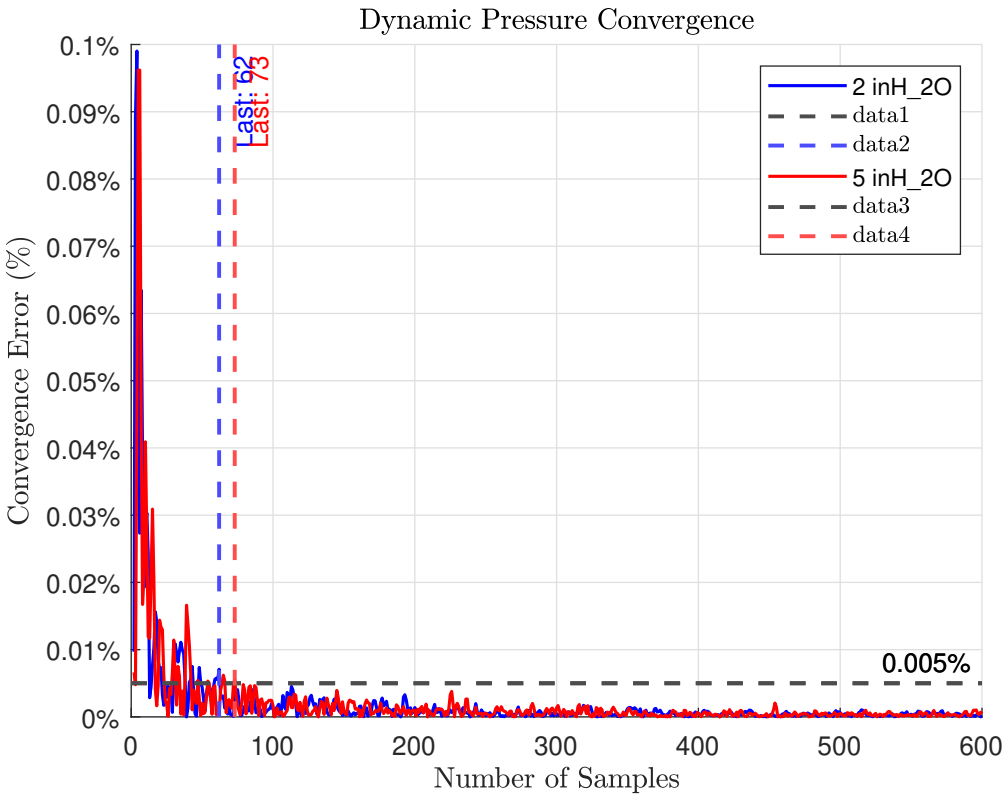

Fig. 16 Convergence test over 600 samples at 2 inH_2O and 5 inH_2O.

Table 2 Convergence Results for Dynamic Pressure

Test Condition	Samples Required for Convergence
2 inH_2O	62 Samples
5 inH_2O	73 Samples

VII. Discussion

The negative pressure readings at Port #7 are an expected behavior due to flow separation occurring at that location. As the Indicated Airspeed (IAS) increases, the velocity difference between the main flow and the recirculating region grows, leading to a further reduction in pressure according to Bernoulli's principle.

The non-zero average pressure reading for IAS = 0 may be attributed to sensor noise, ambient pressure fluctuations, or minor errors in zeroing the pressure transducers. Since the dynamic pressure is computed based on the differential pressure readings, any offset in the baseline measurement could introduce small deviations in Figures 2 and 3. However, these deviations are likely within the acceptable experimental error margins.

The pressure and airspeed measurements at Port #34 exhibit deviations from the expected average values. This could result from localized turbulence, obstruction, or slight misalignment of the Pitot tube. While such discrepancies influence flow uniformity measurements, they do not necessarily invalidate the experiment but highlight the importance of accounting for anomalies and potential test section irregularities.

An increase in air temperature was observed with increasing airspeed. This phenomenon occurs due to adiabatic heating caused by air compression, as well as frictional heating from the tunnel walls. Additionally, interaction between the test section and the surrounding environment contributes to the observed temperature rise.

To mitigate temperature variations, pre-conditioning the incoming air, implementing heat exchangers, or maintaining a controlled test environment can be effective strategies. Reducing test duration and ensuring proper ventilation can further help regulate temperature fluctuations.

The rise in temperature affects both the IAS calculation and the unit Reynolds number. As temperature increases, air density decreases, altering the dynamic pressure readings. Since the Reynolds number is defined as:

$$Re = \frac{\rho V L}{\mu} \tag{7}$$

a reduction in air density leads to a lower Reynolds number, affecting the boundary layer behavior and overall flow characteristics in the test section.

VIII. Conclusion

This experiment analyzed the flow uniformity in a low-speed wind tunnel using a total pressure rake. Dynamic pressure and airspeed variations followed expected trends, validating Bernoulli's equation. Negative pressure at port #7 and deviations at port #34 suggested localized flow disturbances. Temperature effects on IAS and Reynolds number were observed, emphasizing the need for environmental control. Convergence analysis showed 62 and 73 samples were required for 2 inH_2O and 5 inH_2O, respectively. The study reinforced wind tunnel testing principles and the importance of data reduction techniques.

Acknowledgments

The author would like to thank Dr. Xiaofeng Liu for his guidance and Teacher's Assistant Andrew Balolong for assistance during the experiment.

IX. Appendix

A. Sample Data

The following table presents a portion of the raw experimental data collected for the `0inH2O.csv` dataset. Due to space limitations, only the first few rows are shown as a representative sample.

Table 3 Excerpt from `0inH2O.csv` dataset

Port No.	1	2	3	4	5	6	7	8	9	10	11	12
Sample 1	0.000124	0.000048	0.000038	0.000121	0.000145	0.000003	0.000079	0.000057	0.000073	-0.00008	0.000045	-0.000056
Sample 2	0.000138	0.000014	0.000033	0.000133	0.000154	0.000036	0.000065	-0.000008	0.000059	-0.000069	0.000048	-0.00005
Sample 3	0.000126	-0.000022	0.00001	0.000107	0.000126	-0.000006	0.000029	0.000021	0.000031	-0.000115	0.000011	-0.000079
Sample 4	0.000114	-0.000008	0.000024	0.000104	0.00013	-0.00003	0.000023	0.000018	-0.000002	-0.00011	0.000023	-0.000103
Sample 5	0.000119	-0.000001	0.000039	0.000107	0.000139	0.000024	0.000043	0.000006	0.000016	-0.000057	0.000058	-0.000103
Sample 6	0.000118	0.000016	0.000055	0.000078	0.000136	0.000019	0.000033	0.000015	0.000064	-0.00005	0.000048	-0.000066
Sample 7	0.000103	-0.000012	0.00003	0.000066	0.00014	0.000001	0.000036	0.000045	0.000078	-0.00009	0.00002	-0.000071
Sample 8	0.000069	0.00001	-0.000006	0.000103	0.000129	0.00002	0.000048	0.000025	0.000051	-0.000101	0.000023	-0.000076
Sample 9	0.000034	0.000021	-0.000014	0.000079	0.000095	0.000019	0.000054	-0.000017	0.000016	-0.000113	0.000015	-0.000109
Sample 10	0.000079	-0.000007	-0.000016	0.000061	0.000086	-0.000019	0.000024	-0.000008	0.000026	-0.000113	0.000018	-0.000146
Sample 11	0.000097	-0.000026	-0.000026	0.000064	0.000095	-0.000021	0.000002	-0.000031	0.000013	-0.000135	0.000023	-0.000129
Sample 12	0.000092	-0.000013	-0.000031	0.000065	0.000088	-0.000018	0.00002	-0.000014	0.000006	-0.000133	0.000019	-0.000135
Sample 13	0.000068	-0.000012	-0.000033	0.000032	0.000073	-0.000013	0.000021	0.000013	0.000015	-0.000105	0.000022	-0.00013
Sample 14	0.000051	-0.000008	-0.000008	0.000023	0.000082	-0.000029	0.00002	-0.000018	0.000029	-0.000083	0.000017	-0.000066
Sample 15	0.000083	-0.000006	-0.000002	0.000019	0.00013	0.000002	0.00001	0.000008	0.000036	-0.000101	-0.000005	-0.000056
Sample 16	0.000106	-0.00002	0.000025	0.000036	0.00013	0.000024	0.000009	0.00001	0.000031	-0.000104	-0.000016	-0.000062
Sample 17	0.000093	-0.000037	-0.000001	0.000032	0.000081	-0.000001	0.000017	-0.000044	0.000028	-0.000103	-0.000001	-0.000117
Sample 18	0.000073	-0.000018	-0.000045	0.000008	0.000073	-0.000006	0.000005	-0.000085	0.000016	-0.000105	0.000003	-0.000108
Sample 19	0.000048	-0.00002	-0.000053	0.000008	0.000087	-0.000035	0.000029	-0.000059	-0.000001	-0.000124	-0.000031	-0.000102
Sample 20	0.000037	-0.000008	-0.000035	0.000048	0.00009	-0.000024	0.000046	-0.000023	0.000016	-0.000116	-0.000013	-0.000073
Sample 21	0.000054	-0.000019	-0.000031	0.000017	0.00008	-0.000004	0.000041	-0.000017	0.000018	-0.000086	-0.00001	-0.000097
Sample 22	0.000073	-0.00004	-0.000007	0.000011	0.000076	-0.00003	0.000026	-0.000019	-0.000016	-0.000095	-0.000007	-0.000141
Sample 23	0.000057	-0.00007	-0.000013	-0.000003	0.000077	-0.000034	0.00001	-0.000008	-0.000006	-0.000126	0.000004	-0.000114
Sample 24	0.000059	-0.000064	-0.000044	-0.000022	0.000084	-0.000031	0.000005	-0.000034	0.000012	-0.000104	0.00002	-0.000086
Sample 25	0.000039	-0.000045	-0.000076	-0.000003	0.000082	-0.000052	-0.000001	-0.000061	0.000021	-0.000094	0.000006	-0.000094
Sample 26	0.000021	-0.000033	-0.000071	0.000035	0.000062	-0.000065	0.000018	-0.000048	0.000004	-0.000105	-0.000017	-0.000087
Sample 27	0.000039	-0.000045	-0.000058	0.000034	0.000055	-0.000068	0.000017	-0.000005	0.000002	-0.000094	-0.000013	-0.000092
Sample 28	0.000037	-0.000024	-0.000046	0.000043	0.000065	-0.000069	-0.000003	-0.000031	0.000008	-0.00008	-0.000019	-0.000087
Sample 29	0.000051	-0.000028	-0.000064	0.00001	0.000074	-0.000047	0.000004	-0.000023	-0.000012	-0.000086	-0.000046	-0.000107
Sample 30	0.000044	-0.000059	-0.000058	-0.000025	0.000055	-0.000052	0.00002	-0.000031	-0.000008	-0.000077	-0.000017	-0.000113
Sample 31	0.000016	-0.000073	-0.000023	-0.00003	0.000071	-0.000044	0.000047	-0.000032	0.000021	-0.000083	0.00001	-0.000111
Sample 32	0.000047	-0.000046	-0.000026	0.000003	0.000105	-0.000029	0.000035	-0.00001	0.000015	-0.000098	-0.000003	-0.000079
Sample 33	0.000026	-0.000036	-0.000026	-0.000043	0.000093	-0.000029	0.000021	-0.00002	-0.000006	-0.000126	-0.000035	-0.000087
Sample 34	0.000002	-0.00003	-0.000044	-0.000077	0.00007	-0.000067	0.000019	-0.000057	-0.000014	-0.000134	-0.000035	-0.000151
Sample 35	0	-0.000036	-0.00007	-0.00004	0.00005	-0.000085	0.000015	-0.000051	-0.000034	-0.000136	-0.000032	-0.000174
Sample 36	0.000018	-0.000044	-0.000038	-0.000021	0.000064	-0.000069	0.000017	-0.000036	-0.000029	-0.000102	-0.000026	-0.000123
Sample 37	0.000044	-0.00004	-0.000023	-0.000016	0.000073	-0.000076	0.000013	-0.000033	0.000007	-0.000098	-0.000001	-0.000113
Sample 38	0.000047	-0.000023	-0.000044	-0.000009	0.000039	-0.000087	0.000024	-0.000059	0.000008	-0.00011	0.000008	-0.000113
Sample 39	0.000004	-0.00004	-0.000074	-0.00006	0.000006	-0.000061	0.000014	-0.000086	-0.000039	-0.000112	-0.00004	-0.000116
Sample 40	0.000012	-0.000049	-0.000084	-0.000068	-0.000004	-0.000058	-0.000019	-0.000066	-0.000074	-0.000112	-0.000039	-0.000072
Sample 41	-0.000004	-0.000041	-0.000064	-0.00008	0.000014	-0.000068	-0.000026	-0.000038	-0.000088	-0.000094	-0.000035	-0.000095
Sample 42	0.000005	-0.000034	-0.000044	-0.000069	0.000025	-0.000077	-0.000018	-0.000048	-0.000044	-0.000075	-0.000034	-0.000121
Sample 43	0.000006	-0.000074	-0.000085	-0.000041	0.000038	-0.000096	-0.000011	-0.000057	-0.000025	-0.0001	-0.000046	-0.000116
Sample 44	-0.00002	-0.000104	-0.000072	-0.000043	0.000048	-0.000084	-0.000005	-0.000095	-0.000043	-0.000106	-0.000058	-0.00012
Sample 45	-0.000011	-0.00009	-0.000072	-0.000091	0.000035	-0.000078	0.000011	-0.000113	-0.000034	-0.000104	-0.00005	-0.000136
Sample 46	0.000028	-0.000058	-0.000055	-0.000118	0.00003	-0.000098	0.000024	-0.00009	-0.000015	-0.000124	-0.000057	-0.000165
Sample 47	0.000036	-0.000052	-0.000051	-0.000116	0.000028	-0.00006	0.000015	-0.000081	-0.000025	-0.000144	-0.000069	-0.000153
Sample 48	0.000012	-0.000072	-0.000062	-0.000071	0.000034	-0.000099	0.000008	-0.000059	-0.000035	-0.000132	-0.000062	-0.000105
Sample 49	0.000004	-0.000073	-0.000076	-0.000066	0.000005	-0.000137	0.00001	-0.000075	-0.000041	-0.000111	-0.000088	-0.000109
Sample 50	-0.000009	-0.000056	-0.000101	-0.000084	-0.000005	-0.000102	0.000014	-0.000116	-0.000051	-0.000103	-0.000099	-0.000125
Sample 51	0.000015	-0.000083	-0.000091	-0.000072	0.000008	-0.000104	0.000006	-0.000089	-0.000045	-0.000146	-0.0001	-0.000149
Sample 52	0.000043	-0.000087	-0.000071	-0.000047	-0.000017	-0.000099	-0.00002	-0.000042	-0.000037	-0.000152	-0.000066	-0.000143
Sample 53	0.000012	-0.000053	-0.000077	-0.000048	-0.000033	-0.000072	-0.00003	-0.000045	-0.000035	-0.000114	-0.000059	-0.00014
Sample 54	0.000004	-0.000026	-0.000074	-0.000065	-0.000027	-0.000089	-0.000027	-0.000093	-0.000028	-0.000116	-0.000064	-0.000123
Sample 55	0	-0.000053	-0.000068	-0.000082	-0.000004	-0.000102	0.000006	-0.000092	-0.000041	-0.000142	-0.000052	-0.000117
Sample 56	-0.00001	-0.000043	-0.000074	-0.000052	0.000026	-0.000078	0.000002	-0.000075	-0.00006	-0.000104	-0.000069	-0.000093
Sample 57	-0.000019	-0.000053	-0.000083	-0.000072	0.000025	-0.000051	-0.000007	-0.000067	-0.000056	-0.000081	-0.000067	-0.000122
Sample 58	-0.000025	-0.000085	-0.000059	-0.000102	0.000024	-0.000072	-0.000002	-0.000112	-0.000036	-0.000114	-0.000054	-0.000147
Sample 59	-0.00003	-0.000113	-0.000075	-0.000087	0.000026	-0.000109	-0.000031	-0.000119	-0.000047	-0.000156	-0.000089	-0.000138
Sample 60	-0.000016	-0.000126	-0.000096	-0.000083	0.000014	-0.000111	-0.000041	-0.000103	-0.000038	-0.000166	-0.000106	-0.000145
Sample 61	0.000013	-0.000093	-0.000081	-0.000108	0.000014	-0.000111	-0.000027	-0.000094	-0.000034	-0.000182	-0.000089	-0.000167
Sample 62	-0.000001	-0.000091	-0.000059	-0.000104	0.000017	-0.000111	-0.000015	-0.000082	-0.000051	-0.000181	-0.00008	-0.000162
Sample 63	-0.000012	-0.000102	-0.000077	-0.000079	0.000023	-0.000092	0.000005	-0.000093	-0.000036	-0.000149	-0.000068	-0.000141
Sample 64	0.000002	-0.000081	-0.000108	-0.000045	0.000003	-0.000068	-0.000024	-0.000094	-0.000049	-0.000149	-0.000089	-0.000185
Sample 65	-0.000002	-0.000097	-0.000103	-0.000045	0	-0.000085	-0.000058	-0.000089	-0.000061	-0.000208	-0.00012	-0.000223
Sample 66	-0.000029	-0.000106	-0.000083	-0.000056	0.000005	-0.000107	-0.00002	-0.000074	-0.000047	-0.000224	-0.000083	-0.000161
Sample 67	-0.000029	-0.00011	-0.000078	-0.000063	0.000022	-0.000115	0	-0.000071	-0.000042	-0.000182	-0.000052	-0.000094
Sample 68	-0.000022	-0.00009	-0.000088	-0.000084	0.000021	-0.00014	-0.000022	-0.000063	-0.000022	-0.000175	-0.000043	-0.000097
Sample 69	-0.000028	-0.000062	-0.000086	-0.000094	-0.000016	-0.000111	-0.000021	-0.000078	-0.000023	-0.000192	-0.00006	-0.000155
Sample 70	-0.000013	-0.000068	-0.000074	-0.000096	-0.000014	-0.000096	0.000004	-0.000111	-0.000078	-0.00018	-0.000091	-0.000196
Sample 71	-0.000022	-0.000083	-0.000092	-0.00011	-0.000042	-0.000124	0	-0.000088	-0.000102	-0.000173	-0.000094	-0.000206
Sample 72	-0.000016	-0.000075	-0.000089	-0.000097	-0.000044	-0.000116	-0.000021	-0.000052	-0.000074	-0.000174	-0.00008	-0.000185

The complete dataset consists of 600 samples recorded across 37 pressure ports, along with total and static pressure measurements. This data was used for further analysis and flow characterization.

B. MATLAB Code

Listing 1 MATLAB Code for Data Analysis

```
%% AE 303 Lab 2: Test Section Flow Uniformity Characterization
% San Diego State University
% Author: Parham Khodadi
% Date: 02/04/2025

clear; clc; close all;

%% Constants for Conversions
P_ambient_inHg = 30.16; % Ambient pressure in inHg
T_ambient_F = 71.7; % Ambient temperature in Fahrenheit
T_tunnel_F = [78.3731,76.6152,82.7341]; % Wind tunnel temperature in Fahrenheit

% Convert ambient pressure to psi (1 inHg = 0.4912 psi)
P_ambient_psi = P_ambient_inHg * 0.4912;

% Convert temperatures to Kelvin
T_ambient_K = (T_ambient_F - 32) * (5/9) + 273.15;
T_tunnel_K = (T_tunnel_F - 32) * (5/9) + 273.15;

% Conversion Factors
psi_to_Pa = 6894.76;  % Convert psi to Pascals
Pa_to_psi = 1 / psi_to_Pa;
inH2O_to_psi = 0.0360912;

%% Load Pitot Tube Geometry
geometry_file = 'PitotTube_Geometry.csv';
pitot_geometry = readmatrix(geometry_file);

% Extract X and Z coordinates for plotting
port_numbers = pitot_geometry(:, 1);
x_coords = pitot_geometry(:, 2);
z_coords = pitot_geometry(:, 3);

fprintf('Loaded Pitot Tube Geometry: %d ports.\n', length(port_numbers));

%% Compute Indicated Airspeed (IAS) for Each Test Condition

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};

% Constants
rho_air = 1.225; % Air density in kg/m^3 (standard conditions)
psi_to_Pa = 6894.76; % Conversion factor from psi to Pascals

% Initialize storage for IAS values
IAS_values = zeros(length(files), 1);
IAS_values(1) = 0;
```

```
for i = 2:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Extract total and static pressures from Ports 61 & 62
    P_static_psi = data(2:601, 37);  % Static pressure at test section inlet (Port 61)
    P_total_psi = data(2:601, 38);   % Total pressure at test section inlet (Port 62)

    % Compute dynamic pressure at test section inlet (psi)
    q_psi = P_total_psi - P_static_psi;

    % Convert dynamic pressure to Pascals
    q_Pa = q_psi * psi_to_Pa;

    % Compute IAS using Bernoulli's equation: V = sqrt(2q/rho)
    airspeed_mps = sqrt(2 * q_Pa / rho_air);

    % Compute mean IAS across all 600 samples
    IAS_values(i) = mean(airspeed_mps);

    % Display results
    fprintf('Indicated Airspeed (IAS) for %s: %.3f m/s\n', files{i}, IAS_values(i));
end

% Save results for later use in analysis
save('IAS_Values.mat', 'IAS_values');


%% Objective #1 - Load Data and Compute Mean Dynamic Pressure

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};

% Initialize storage for mean dynamic pressures
mean_dynamic_pressure = zeros(length(files), 1);

% Loop through each file
for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Compute dynamic pressure: q = P_total - P_static
    dynamic_pressure_psi = data(2:601, 38) - data(2:601, 37);  % Now in psi

    % Compute mean dynamic pressure across all ports
    mean_dynamic_pressure(i) = mean(dynamic_pressure_psi(:));

    % Display results
    fprintf('Mean Dynamic Pressure for %s: %.3f psi\n', files{i}, mean_dynamic_pressure(i)
end

% Save results
save('Mean_Dynamic_Pressure.mat', 'mean_dynamic_pressure');
```

```
%% Objective #2 - Unit Reynolds Number Calculation
% This section computes the unit Reynolds number for each wind tunnel speed.

% Constants
R_specific = 287.05; % Specific gas constant for air (J/kg K)
mu_air = 1.846e-5;  % Dynamic viscosity of air at standard conditions (Pa s)
m_to_in = 1 / 39.3701; % Conversion factor from meters to inches

% Initialize storage for Reynolds numbers
unit_reynolds_numbers = zeros(length(files), 1);

% Compute Reynolds number for each condition
for i = 1:length(files)
    % Compute air density using the ideal gas law: rho = P / (R * T)
    rho_air = (P_ambient_psi * psi_to_Pa) / (R_specific * T_tunnel_K(i)); % in kg/m

    % Compute freestream velocity: V = sqrt(2q / rho)
    V_inf = sqrt(2 * (mean_dynamic_pressure(i) * psi_to_Pa) / rho_air); % in m/s

    % Compute unit Reynolds number in 1/m
    unit_reynolds_numbers_m = (rho_air * V_inf) / mu_air; % in 1/m

    % Convert to 1/in
    unit_reynolds_numbers(i) = unit_reynolds_numbers_m * m_to_in; % in 1/in

    % Display results
    fprintf('Unit␣Reynolds␣Number␣for␣%s:␣Re_L␣=␣%.3e␣(1/in)\n', files{i}, unit_reynolds_nu
end

% Save results
save('Unit_Reynolds_Number.mat', 'unit_reynolds_numbers');

%% Objective #3 - Data Reduction & Analysis
%% Figure 1 - Averaged Pressure for Three Runs (Scatter Plot + Horizontal Averages)
figure;
hold on;

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};
colors = {[0, 0.4470, 0.7410], [0.8500, 0.3250, 0.0980], [0.9290, 0.6940, 0.1250]}; % Blue
legend_labels = {'$0.0$␣inH$_{2}$O', '$2.0$␣inH$_{2}$O', '$5.0$␣inH$_{2}$O'}; % Correct La

% Initialize storage for overall average pressure per dataset
all_pressures = zeros(length(files), 35); % Assuming 35 ports

for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Extract gauge pressure readings from ports 1-35 (columns 2-36)
    gauge_pressure_psi = data(2:601, 2:36);
```

```
    % Compute mean pressure per port across all samples (600 samples)
    avg_pressure = mean(gauge_pressure_psi, 1);

    % Store for overall averaging
    all_pressures(i, :) = avg_pressure;

    % Scatter plot for individual pressure measurements (no lines)
    scatter(1:35, avg_pressure, 40, colors{i}, 'filled', 'DisplayName', legend_labels{i});

    % Plot a horizontal black line for the average pressure of each dataset
    yline(mean(avg_pressure), 'k-', 'LineWidth', 1.5);
end

xlabel('Pressure Port', 'Interpreter', 'Latex');
ylabel('Pressure (psi)', 'Interpreter', 'Latex');
title('Flow Uniformity - Measured Pressure', 'Interpreter', 'Latex');
legend('Location', 'northeast', 'Interpreter', 'Latex');
grid on;
hold off;
print -depsc Fig1.eps


%% Figure 2 - Averaged Measured Dynamic Pressure for Three Runs
figure;
hold on;

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};
colors = {[0, 0.4470, 0.7410], [0.8500, 0.3250, 0.0980], [0.9290, 0.6940, 0.1250]}; % Blue
legend_labels = {'$0.0$ inH$_{2}$O', '$2.0$ inH$_{2}$O', '$5.0$ inH$_{2}$O', 'average', 'q

% Initialize storage for dynamic pressure data
all_dynamic_pressures = zeros(length(files), 35); % Assuming 35 ports
scatter_handles = gobjects(length(files), 1); % Store scatter handles for legend

for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Extract total and static pressures **per port**
    P_total_psi = data(2:601, 2:36);  % Total pressure from ports 1-35
    P_static_psi = data(2:601, 37);   % Static pressure (single column)

    % Compute dynamic pressure for each port
    dynamic_pressure_psi = P_total_psi - P_static_psi; % Element-wise subtraction

    % Compute mean dynamic pressure per port across all samples (600 samples)
    avg_dynamic_pressure = mean(dynamic_pressure_psi, 1);

    % Store for overall averaging
    all_dynamic_pressures(i, :) = avg_dynamic_pressure;

    % Scatter plot for individual dynamic pressure measurements
    scatter(1:35, avg_dynamic_pressure, 40, colors{i}, 'o', 'filled'); % Plot points
```

```
    % Create a **separate scatter object for the legend** (only one per dataset)
    scatter_handles(i) = scatter(NaN, NaN, 40, colors{i}, 'o', 'filled'); % Invisible poin

    % Plot a horizontal black line for the average dynamic pressure of each dataset
    avg_line = yline(mean(avg_dynamic_pressure), 'k-', 'LineWidth', 1.5);
end

% Plot a horizontal red line for the q_setting of each dataset
q_lines = [
    yline(0, 'r-', 'LineWidth', 1.5);
    yline(2 * inH2O_to_psi, 'r-', 'LineWidth', 1.5);
    yline(5 * inH2O_to_psi, 'r-', 'LineWidth', 1.5)
];

% Create dummy scatter objects for legend entries "average" and "q_setting"
scatter_avg = scatter(NaN, NaN, 40, 'k', 's', 'filled'); % Black square for "average"
scatter_qsetting = scatter(NaN, NaN, 40, 'r', 's', 'filled'); % Red square for "q_setting

% Set legend with scatter handles + dummy markers for average and q_setting
legend([scatter_handles; scatter_avg; scatter_qsetting], legend_labels, 'Location', 'north

xlabel('Pressure␣Port', 'Interpreter', 'Latex');
ylabel('Pressure␣(psi)', 'Interpreter', 'Latex');
title('Flow␣Uniformity␣-␣Dynamic␣Pressure', 'Interpreter', 'Latex');
grid on;
hold off;
print -depsc Fig2.eps

%% Figure 3 - Test Section Airspeed
figure;
hold on;

load('IAS_Values.mat', 'IAS_values');

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};
colors = {[0, 0.4470, 0.7410], [0.8500, 0.3250, 0.0980], [0.9290, 0.6940, 0.1250]}; % Blue
legend_labels = {sprintf('%.1f␣m/s', IAS_values(1)), sprintf('%.1f␣m/s', IAS_values(2)), s

rho_air = 1.225; % Air density in kg/m^3 (standard conditions)

% Initialize storage for computed airspeeds
all_airspeeds = zeros(length(files), 35); % Assuming 35 ports
scatter_handles = gobjects(length(files), 1); % Store scatter handles for legend

for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Extract total and static pressures **per port** (already in psi)
    P_total_psi = data(2:601, 2:36);  % Total pressure from ports 1-35
    P_static_psi = data(2:601, 37);   % Static pressure (single column)
```

```
    % Compute dynamic pressure for each port (psi)
    q_psi = P_total_psi - P_static_psi; % Bernoulli: q = P_total - P_static

    % Convert dynamic pressure from psi to Pascals
    q_Pa = q_psi * psi_to_Pa;

    % Fix: Ensure dynamic pressure is non-negative to avoid complex values
    q_Pa = max(q_Pa, 0); % Clamp negative values to zero

    % Compute airspeed using Bernoulli's equation: V = sqrt(2q/rho)
    airspeed_mps = sqrt(2 * q_Pa / rho_air);

    % Compute mean airspeed per port across all samples (600 samples)
    avg_airspeed = mean(airspeed_mps, 1);

    % Store for overall averaging
    all_airspeeds(i, :) = avg_airspeed;

    % Scatter plot for measured airspeed at each port
    scatter_handles(i) = scatter(1:35, avg_airspeed, 40, colors{i}, 'o', 'filled'); % Plot
end

% Plot horizontal black line for the average airspeed of each dataset
for i = 1:length(files)
    if all(isreal(all_airspeeds(i, :))) % Ensure all values are real
        yline(mean(all_airspeeds(i, :)), 'k-', 'LineWidth', 1.5);
    end
end

% Plot red horizontal lines for indicated airspeed (IAS) settings
q_setting_lines = [
    yline(IAS_values(1), 'r-', 'LineWidth', 1.5);
    yline(IAS_values(2), 'r-', 'LineWidth', 1.5);
    yline(IAS_values(3), 'r-', 'LineWidth', 1.5)
];

% Create dummy scatter objects for legend entries "average" and "V_setting"
scatter_avg = scatter(NaN, NaN, 40, 'k', 's', 'filled'); % Black square for "average"
scatter_qsetting = scatter(NaN, NaN, 40, 'r', 's', 'filled'); % Red square for "V_setting

% Set legend with the stored representative scatter handles
legend([scatter_handles; scatter_avg; scatter_qsetting], legend_labels, 'Location', 'north

xlabel('Pressure Port', 'Interpreter', 'Latex');
ylabel('Airspeed (m/s)', 'Interpreter', 'Latex');
title('Flow Uniformity - Test Section Airspeed', 'Interpreter', 'Latex');
grid on;
hold off;
print -depsc Fig3.eps

%% Figure 4 - Dynamic Pressure Deviation from Average
figure;
hold on;
```

```
% Define datasets (Only 2 inH2O and 5 inH2O)
test_conditions = {'2inH2O.csv', '5inH2O.csv'};
colors = {[0, 0, 0], [1, 0, 0]}; % Black for 2 inH2O, Red for 5 inH2O
legend_labels = {'$2.0$ inH$_{2}$O', '$5.0$ inH$_{2}$O'}; % LaTeX format

% Load mean dynamic pressure values from Objective 1
load('Mean_Dynamic_Pressure.mat', 'mean_dynamic_pressure');

% Define port numbers (excluding Port 7)
ports = 1:35;
ports(7) = []; % Remove Port 7 (static pressure probe)

% Initialize storage for deviations
dq_values = zeros(length(test_conditions), length(ports));

for i = 1:length(test_conditions)
    % Read CSV file
    data = readmatrix(test_conditions{i});

    % Extract total and static pressures for ports 1–35
    P_total_psi = data(2:601, 2:36);  % Total pressure (ports 1–35)
    P_static_psi = data(2:601, 37);   % Static pressure (single column)

    % Compute dynamic pressure: q = P_total - P_static
    dynamic_pressure_psi = P_total_psi - P_static_psi;

    % Compute mean dynamic pressure per port
    avg_dynamic_pressure = mean(dynamic_pressure_psi, 1);

    % Compute deviation from **mean dynamic pressure per port**
    deviation = 100 * (avg_dynamic_pressure - mean_dynamic_pressure(i+1)) ./ mean_dynamic_

    deviation(7) = [];
    % Store for plotting
    dq_values(i, :) = deviation;
end

% Ensure correct data dimensions
if size(dq_values, 2) ~= length(ports)
    error('Mismatch in vector sizes! Check ports and computed deviations.');
end

% Plot deviations
scatter(ports, dq_values(1, :), 50, 'MarkerEdgeColor', colors{1}, 'Marker', 'o', 'MarkerFa
scatter(ports, dq_values(2, :), 50, 'MarkerEdgeColor', colors{2}, 'Marker', 'd', 'MarkerFa

% Set legend
legend(legend_labels, 'Location', 'southeast', 'Interpreter', 'Latex');

% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
ylabel('$\frac{q - \bar{q}}{\bar{q}} \times 100\%$', 'Interpreter', 'Latex');
title('Flow Uniformity - Dynamic Pressure Deviation From Average', 'Interpreter', 'Latex')
```

```
% Formatting
ytickformat('percentage');
xlim([1 35]);
ylim([-5 2.5]); % Match expected range
xticks(1:2:35);
yticks(-5:0.5:2.5);
grid on;
hold off;
print -depsc Fig4.eps

%% Figure 5 - Dynamic Pressure Deviation for 2 inH2O
figure;
hold on;

% Load mean dynamic pressure values from Objective 1
load('Mean_Dynamic_Pressure.mat', 'mean_dynamic_pressure');

% Define dataset and properties
test_condition = '2inH2O.csv';
color = [0, 0, 0]; % Black
marker = 'o'; % Circle
legend_label = '$2.0$ inH$_{2}$O'; % LaTeX format

% Define port numbers (excluding Port 7)
ports = 1:35;
ports(7) = []; % Remove Port 7 (static pressure probe)

% Read CSV file
data = readmatrix(test_condition);

% Extract total and static pressures
P_total_psi = data(2:601, 2:36);  % Total pressure (ports 1-35)
P_static_psi = data(2:601, 37);   % Static pressure (single column)

% Compute dynamic pressure: q = P_total - P_static
dynamic_pressure_psi = P_total_psi - P_static_psi;

% Compute mean dynamic pressure per port
avg_dynamic_pressure = mean(dynamic_pressure_psi, 1);

% Compute deviation from **mean dynamic pressure per port**
deviation = 100 * (avg_dynamic_pressure - mean_dynamic_pressure(2)) ./ mean_dynamic_pressu

% Remove Port 7 data
deviation(7) = [];

% Scatter plot
scatter(ports, deviation, 50, 'MarkerEdgeColor', color, 'Marker', marker, 'MarkerFaceColor

% Set legend
legend(legend_label, 'Location', 'southeast', 'Interpreter', 'Latex');

% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
```

```
ylabel('$\frac{q - \bar{q}}{\bar{q}} \times 100\%$', 'Interpreter', 'Latex');
title('Flow Uniformity - Dynamic Pressure Deviation from Average 2.0 inH$_2$O', 'Interprete

% Formatting
ytickformat('percentage');
xlim([1 35]);
ylim([-5 2.5]); % Match expected range
xticks(1:2:35);
yticks(-5:0.5:2.5);
grid on;
hold off;
print -depsc Fig5.eps

%% Figure 6 - Dynamic Pressure Deviation for 5 inH2O
figure;
hold on;

% Define dataset and properties
test_condition = '5inH2O.csv';
color = [1, 0, 0]; % Red
marker = 'd'; % Diamond
legend_label = '$5.0$ inH$_{2}$O'; % LaTeX format

% Read CSV file
data = readmatrix(test_condition);

% Extract total and static pressures
P_total_psi = data(2:601, 2:36);  % Total pressure (ports 1-35)
P_static_psi = data(2:601, 37);   % Static pressure (single column)

% Compute dynamic pressure: q = P_total - P_static
dynamic_pressure_psi = P_total_psi - P_static_psi;

% Compute mean dynamic pressure per port
avg_dynamic_pressure = mean(dynamic_pressure_psi, 1);

% Compute deviation from **mean dynamic pressure per port**
deviation = 100 * (avg_dynamic_pressure - mean_dynamic_pressure(3)) ./ mean_dynamic_pressu

% Remove Port 7 data
deviation(7) = [];

% Scatter plot
scatter(ports, deviation, 50, 'MarkerEdgeColor', color, 'Marker', marker, 'MarkerFaceColor

% Set legend
legend(legend_label, 'Location', 'southeast', 'Interpreter', 'Latex');

% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
ylabel('$\frac{q - \bar{q}}{\bar{q}} \times 100\%$', 'Interpreter', 'Latex');
title('Flow Uniformity - Dynamic Pressure Deviation from Average 5.0 inH$_2$O', 'Interprete

% Formatting
```

```
ytickformat('percentage');
xlim([1 35]);
ylim([-5 2.5]); % Match expected range
xticks(1:2:35);
yticks(-5:0.5:2.5);
grid on;
hold off;
print -depsc Fig6.eps


%% Figures 7-9 - Compute and Save Mean Airspeed

% Define test condition files
files = {'0inH2O.csv', '2inH2O.csv', '5inH2O.csv'};

% Constants
rho_air = 1.225; % Air density in kg/m^3 (standard conditions)
psi_to_Pa = 6894.76; % Convert psi to Pascals

% Initialize storage for mean airspeed
mean_airspeed = zeros(length(files), 1);

% Loop through each test condition
for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Compute dynamic pressure: q = P_total - P_static
    dynamic_pressure_psi = data(2:601, 38) - data(2:601, 37); % q in psi

    % Convert dynamic pressure to Pascals
    q_Pa = dynamic_pressure_psi * psi_to_Pa;

    % Ensure q is non-negative to prevent complex numbers
    q_Pa = max(q_Pa, 0);

    % Compute airspeed using Bernoulli's equation: V = sqrt(2q / rho)
    airspeed_mps = sqrt(2 * q_Pa / rho_air);

    % Compute mean airspeed across all ports
    mean_airspeed(i) = mean(airspeed_mps(:));

    % Display results
    fprintf('Mean Airspeed for %s: %.3f m/s\n', files{i}, mean_airspeed(i));
end

% Save results for later use
save('Mean_Airspeed.mat', 'mean_airspeed');

%% Figure 7 - Airspeed Deviation for 2 inH2O and 5 inH2O
figure;
hold on;

% Define test conditions (2 inH2O and 5 inH2O)
```

```
test_conditions = {'2inH2O.csv', '5inH2O.csv'};
colors = {[0, 0, 0], [1, 0, 0]}; % Black for 2 inH2O, Red for 5 inH2O
markers = {'o', 'd'}; % Circle for 2 inH2O, Diamond for 5 inH2O
legend_labels = {'$2.0$␣inH$_{2}$O', '$5.0$␣inH$_{2}$O'}; % LaTeX format

% Load mean airspeed values
load('Mean_Airspeed.mat', 'mean_airspeed');

% Define port numbers (excluding Port 7)
ports = 1:35;
ports(7) = []; % Remove Port 7 (static pressure probe)

% Initialize storage for airspeed deviations
V_deviation = zeros(length(test_conditions), length(ports));

for i = 1:length(test_conditions)
    % Read CSV file
    data = readmatrix(test_conditions{i});

    % Extract total and static pressures for ports 1–35
    P_total_psi = data(2:601, 2:36);  % Total pressure (ports 1–35)
    P_static_psi = data(2:601, 37);   % Static pressure (single column)

    % Compute dynamic pressure: q = P_total - P_static
    dynamic_pressure_psi = P_total_psi - P_static_psi;

    % Convert dynamic pressure to Pascals
    q_Pa = dynamic_pressure_psi * psi_to_Pa;

    % Ensure q is non-negative
    q_Pa = max(q_Pa, 0);

    % Compute airspeed using Bernoullis equation
    airspeed = sqrt(2 * q_Pa / rho_air);

    % Compute mean airspeed per port
    avg_airspeed = mean(airspeed, 1);

    % Compute deviation from **mean airspeed per port**
    deviation = 100 * (avg_airspeed - mean_airspeed(i+1)) ./ mean_airspeed(i+1);

    % Remove Port 7 data
    deviation(7) = [];

    % Store for plotting
    V_deviation(i, :) = deviation;
end

% Scatter plot for 2 inH2O and 5 inH2O
scatter(ports, V_deviation(1, :), 50, 'MarkerEdgeColor', colors{1}, 'Marker', 'o', 'Marker
scatter(ports, V_deviation(2, :), 50, 'MarkerEdgeColor', colors{2}, 'Marker', 'd', 'Marker

% Set legend
legend(legend_labels, 'Location', 'southeast', 'Interpreter', 'Latex');
```

```
% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
ylabel('$\frac{V - \bar{V}}{\bar{V}} \times 100$ (\%)', 'Interpreter', 'Latex');
title('Flow Uniformity - Airspeed Deviation from Average', 'Interpreter', 'Latex');

% Formatting
ytickformat('percentage');
xlim([1 35]);
ylim([-2.5 1.5]); % Match expected range
xticks(1:2:35);
yticks(-2.5:0.5:1.5);
grid on;
hold off;
print -depsc Fig7.eps

%% Figure 8 - Airspeed Deviation for 2 inH2O
figure;
hold on;

% Scatter plot for 2 inH2O
scatter(ports, V_deviation(1, :), 50, 'MarkerEdgeColor', colors{1}, 'Marker', 'o', 'Marker

% Set legend
legend(legend_labels{1}, 'Location', 'southeast', 'Interpreter', 'Latex');

% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
ylabel('$\frac{V - \bar{V}}{\bar{V}} \times 100$ (\%)', 'Interpreter', 'Latex');
title('Flow Uniformity - Airspeed Deviation from Average 2.0 inH$_2$O', 'Interpreter', 'La

% Formatting
ytickformat('percentage');
xlim([1 35]);
ylim([-2.5 1.5]); % Match expected range
xticks(1:2:35);
yticks(-2.5:0.5:1.5);
grid on;
hold off;
print -depsc Fig8.eps

%% Figure 9 - Airspeed Deviation for 5 inH2O
figure;
hold on;

% Scatter plot for 5 inH2O
scatter(ports, V_deviation(2, :), 50, 'MarkerEdgeColor', colors{2}, 'Marker', 'd', 'Marker

% Set legend
legend(legend_labels{2}, 'Location', 'southeast', 'Interpreter', 'Latex');

% Set labels and title
xlabel('Pressure Port Number', 'Interpreter', 'Latex');
ylabel('$\frac{V - \bar{V}}{\bar{V}} \times 100$ (\%)', 'Interpreter', 'Latex');
```

```
title('Flow Uniformity – Airspeed Deviation from Average 5.0 inH$_2$O', 'Interpreter', 'La

% Formatting
ytickformat('percentage');
xlim([1 35]);
ylim([-2.5 1.5]); % Match expected range
xticks(1:2:35);
yticks(-2.5:0.5:1.5);
grid on;
hold off;
print -depsc Fig9.eps

%% Figures 10 & 11 - Dynamic Pressure Deviation Contour Maps

% Load Pitot Tube Geometry
geometry_file = 'PitotTube_Geometry.csv';
pitot_geometry = readmatrix(geometry_file);
port_numbers = pitot_geometry(:, 1);
x_coords = pitot_geometry(:, 2);
z_coords = pitot_geometry(:, 3);

% Remove Port 7 from dataset
valid_ports = port_numbers ~= 7;
x_coords = x_coords(valid_ports);
z_coords = z_coords(valid_ports);
port_numbers = port_numbers(valid_ports);

% Define test condition files (2 inH2O and 5 inH2O)
files = {'2inH2O.csv', '5inH2O.csv'};
titles = {...
    'Dynamic pressure deviation from average $q=2.0$ inH$_2$O', ...
    'Dynamic pressure deviation from average $q=5.0$ inH$_2$O' ...
};
colorbar_labels = '$\frac{q - \bar{q}}{\bar{q}} \times 100\%$';

% Load mean dynamic pressure from Objective #1
load('Mean_Dynamic_Pressure.mat', 'mean_dynamic_pressure');

% Create a grid for interpolation
xq = linspace(min(x_coords), max(x_coords), 50);
zq = linspace(min(z_coords), max(z_coords), 50);
[Xq, Zq] = meshgrid(xq, zq);

for fig = 1:2
    % Read pressure data
    data = readmatrix(files{fig});

    % Extract total and static pressures for ports 1-35
    P_total_psi = data(2:601, 2:36);  % Total pressure from ports 1-35
    P_static_psi = data(2:601, 37);   % Static pressure (single column)

    % Compute dynamic pressure per port
    dynamic_pressure_psi = P_total_psi - P_static_psi;
```

```
    % Compute mean dynamic pressure per port
    avg_dynamic_pressure = mean(dynamic_pressure_psi, 1);

    % Compute dynamic pressure deviation from mean
    deviation = 100 * (avg_dynamic_pressure - mean_dynamic_pressure(fig+1)) ./ mean_dynam

    % Remove Port 7 from deviation data
    deviation = deviation(valid_ports);

    % Interpolate data onto grid for smooth contour plot
    deviation_interp = griddata(x_coords, z_coords, deviation, Xq, Zq, 'cubic');

    % Create new figure window for each test condition
    figure;
    hold on;

    % Contour plot with black stroke between levels
    contourf(Xq, Zq, deviation_interp, 20, 'LineColor', 'k'); % Add black strokes
    scatter(x_coords, z_coords, 80, 'r', 'filled', 'MarkerEdgeColor', 'k', 'LineWidth', 1.

    % Formatting
    colorbar_handle = colorbar;
    title(titles{fig}, 'Interpreter', 'Latex', 'FontSize', 14);
    xlabel('x␣position␣[in]', 'Interpreter', 'Latex', 'FontSize', 12);
    ylabel('z␣position␣[in]', 'Interpreter', 'Latex', 'FontSize', 12);

    % Correct LaTeX formatting for colorbar label
    colorbar_handle.Label.String = colorbar_labels;
    colorbar_handle.Label.Interpreter = 'Latex';
    colorbar_handle.Label.FontSize = 12;

    hold off;
    print(sprintf('Fig%i.eps', 9+fig),'-depsc')
end


%% Figure 12 - Airspeed Contour Map for Highest IAS Setting

% Load Pitot Tube Geometry
geometry_file = 'PitotTube_Geometry.csv';
pitot_geometry = readmatrix(geometry_file);
port_numbers = pitot_geometry(:, 1);
x_coords = pitot_geometry(:, 2);
z_coords = pitot_geometry(:, 3);

% Remove Port 7 from dataset
valid_ports = port_numbers ~= 7;
x_coords = x_coords(valid_ports);
z_coords = z_coords(valid_ports);
port_numbers = port_numbers(valid_ports);

% Load Indicated Airspeed (IAS) values
load('IAS_Values.mat', 'IAS_values');
IAS_target = IAS_values(3); % Third IAS value (highest setting)
```

```
% Load the 5inH2O dataset (corresponding to highest IAS)
file = '5inH2O.csv';
data = readmatrix(file);

% Extract total and static pressures for ports 1–35
P_total_psi = data(2:601, 2:36);  % Total pressure from ports 1–35
P_static_psi = data(2:601, 37);   % Static pressure (single column)

% Compute dynamic pressure per port
dynamic_pressure_psi = P_total_psi - P_static_psi;

% Constants
rho_air = 1.225; % Air density in kg/m^3 (standard conditions)
psi_to_Pa = 6894.76; % Conversion factor from psi to Pascals

% Convert dynamic pressure to Pascals
q_Pa = dynamic_pressure_psi * psi_to_Pa;

% Compute airspeed using Bernoulli's equation: V = sqrt(2q/rho)
airspeed_mps = sqrt(2 * q_Pa / rho_air);

% Compute mean airspeed per port
avg_airspeed = mean(airspeed_mps, 1);

% Remove Port 7 from airspeed data
avg_airspeed = avg_airspeed(valid_ports);

% Create a grid for interpolation
xq = linspace(min(x_coords), max(x_coords), 50);
zq = linspace(min(z_coords), max(z_coords), 50);
[Xq, Zq] = meshgrid(xq, zq);

% Interpolate airspeed data onto grid
airspeed_interp = griddata(x_coords, z_coords, avg_airspeed, Xq, Zq, 'cubic');

% Create figure
figure;
hold on;

% Contour plot with black stroke between levels
contourf(Xq, Zq, airspeed_interp, 20, 'LineColor', 'k'); % Add black strokes
scatter(x_coords, z_coords, 80, 'r', 'filled', 'MarkerEdgeColor', 'k', 'LineWidth', 1.5); %

% Formatting
colorbar_handle = colorbar;
title(sprintf('Airspeed distribution - IAS=%.1f m/s', IAS_target), 'Interpreter', 'Latex',
xlabel('x position [in]', 'Interpreter', 'Latex', 'FontSize', 12);
ylabel('z position [in]', 'Interpreter', 'Latex', 'FontSize', 12);

% Correct LaTeX formatting for colorbar label
colorbar_handle.Label.String = 'Measured Airspeed (m/s)';
colorbar_handle.Label.Interpreter = 'Latex';
colorbar_handle.Label.FontSize = 12;
```

```
hold off;
print -depsc Fig12.eps

%% Figure 13 - Airspeed Contour Map for 2 inH2O Setting

% Load Pitot Tube Geometry
geometry_file = 'PitotTube_Geometry.csv';
pitot_geometry = readmatrix(geometry_file);
port_numbers = pitot_geometry(:, 1);
x_coords = pitot_geometry(:, 2);
z_coords = pitot_geometry(:, 3);

% Remove Port 7 from dataset
valid_ports = port_numbers ~= 7;
x_coords = x_coords(valid_ports);
z_coords = z_coords(valid_ports);
port_numbers = port_numbers(valid_ports);

% Load Indicated Airspeed (IAS) values
load('IAS_Values.mat', 'IAS_values');
IAS_target = IAS_values(2); % Second IAS value (corresponding to 2 inH2O)

% Load the 2inH2O dataset
file = '2inH2O.csv';
data = readmatrix(file);

% Extract total and static pressures for ports 1-35
P_total_psi = data(2:601, 2:36);  % Total pressure from ports 1-35
P_static_psi = data(2:601, 37);   % Static pressure (single column)

% Compute dynamic pressure per port
dynamic_pressure_psi = P_total_psi - P_static_psi;

% Constants
rho_air = 1.225; % Air density in kg/m^3 (standard conditions)
psi_to_Pa = 6894.76; % Conversion factor from psi to Pascals

% Convert dynamic pressure to Pascals
q_Pa = dynamic_pressure_psi * psi_to_Pa;

% Compute airspeed using Bernoulli's equation: V = sqrt(2q/rho)
airspeed_mps = sqrt(2 * q_Pa / rho_air);

% Compute mean airspeed per port
avg_airspeed = mean(airspeed_mps, 1);

% Remove Port 7 from airspeed data
avg_airspeed = avg_airspeed(valid_ports);

% Create a grid for interpolation
xq = linspace(min(x_coords), max(x_coords), 50);
zq = linspace(min(z_coords), max(z_coords), 50);
[Xq, Zq] = meshgrid(xq, zq);
```

```
% Interpolate airspeed data onto grid
airspeed_interp = griddata(x_coords, z_coords, avg_airspeed, Xq, Zq, 'cubic');

% Create figure
figure;
hold on;

% Contour plot with black stroke between levels
contourf(Xq, Zq, airspeed_interp, 20, 'LineColor', 'k'); % Black strokes for levels
scatter(x_coords, z_coords, 80, 'r', 'filled', 'MarkerEdgeColor', 'k', 'LineWidth', 1.5); %

% Formatting
colorbar_handle = colorbar;
title(sprintf('Airspeed distribution - IAS=%.1f m/s', IAS_target), 'Interpreter', 'Latex',
xlabel('x position [in]', 'Interpreter', 'Latex', 'FontSize', 12);
ylabel('z position [in]', 'Interpreter', 'Latex', 'FontSize', 12);

% Correct LaTeX formatting for colorbar label
colorbar_handle.Label.String = 'Measured Airspeed (m/s)';
colorbar_handle.Label.Interpreter = 'Latex';
colorbar_handle.Label.FontSize = 12;

hold off;
print -depsc Fig13.eps


%% Objective #4 - Convergence Test for Time-Averaged Dynamic Pressure

% Define test condition files
files = {'2inH2O.csv', '5inH2O.csv'};
colors = {'b', 'r'}; % Blue for 2inH2O, Red for 5inH2O
labels = {'2 inH_2O', '5 inH_2O'};
threshold = 0.005 / 100; % Convert 0.005% to decimal

figure;
hold on;

for i = 1:length(files)
    % Read CSV file
    data = readmatrix(files{i});

    % Compute dynamic pressure (q = P_total - P_static)
    q_psi = data(2:601, 38) - data(2:601, 37); % Dynamic pressure from all 600 samples

    % Compute cumulative mean at each sample index
    cumulative_mean = cumsum(q_psi) ./ (1:length(q_psi))';

    % Compute convergence criteria: |q_avg(n) - q_avg(n-1)| / q_avg(n-1)
    convergence_error = abs(diff(cumulative_mean) ./ cumulative_mean(2:end)) * 100; % Conv

    % Find the **last** sample index where error exceeds threshold
    last_convergence_index = find(convergence_error > threshold * 100, 1, 'last') + 1;
```

```
    % If no valid index is found, assume full sample range
    if isempty(last_convergence_index)
        last_convergence_index = length(q_psi);
    end

    % Plot convergence curve
    plot(2:length(q_psi), convergence_error, 'Color', colors{i}, 'LineWidth', 1.2, 'Display

    % Plot horizontal threshold line
    yline(threshold * 100, '--k', 'LineWidth', 1.5, 'Label', '0.005%', 'Interpreter', 'Late

    % Plot vertical marker for the last convergence sample
    xline(last_convergence_index, '--', 'Color', colors{i}, 'LineWidth', 1.5, 'Label', spr

    % Display result
    fprintf('Dynamic pressure converges for %s at %d samples.\n', files{i}, last_convergenc
end

% Formatting
xlabel('Number of Samples', 'Interpreter', 'Latex');
ylabel('Convergence Error (\%)', 'Interpreter', 'Latex');
title('Dynamic Pressure Convergence', 'Interpreter', 'Latex');
legend('show', 'Location', 'northeast', 'Interpreter', 'Latex');
ytickformat('percentage'); % Format y-axis in %
grid on;
hold off;
print -depsc Fig14.eps
```