An Investigation of the Odor-Sensing Abilities of Moths

Submitted by Nicole Sharp
in partial completion of
requirements for
EMAE 398

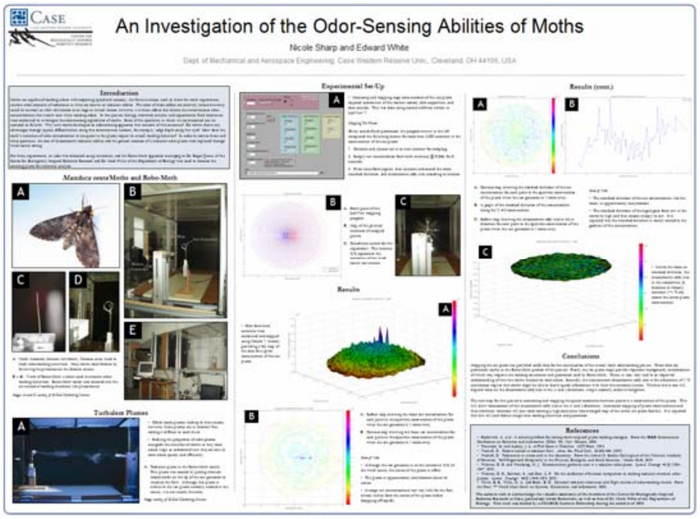
10 May 2006

Advisor: Dr. Edward White

Table of Contents

Introduction

This report describes the essentials of a student project intended to investigate how moths track odors by exploring the fluid mechanics involved. Although the project began in June 2005, equipment problems prevented much from being accomplished until the start of 2006, when I continued the project as my senior capstone project. At the beginning of the Spring 2006 semester, the following deliverables were chosen as goals: custom LabView software to automate experiments; the results of a single-sensor experiment; the results of a double-sensor experiment; and a research poster for the 2006 SOURCE Symposium and Poster Session. As of the writing of this report, all have been completed except the double-sensor experiment, which is under way and discussed later.


An Investigation of the Odor-Sensing Abilities of Moths

Nicole Sharp and Edward White

Dept. of Mechanical and Aerospace Engineering, Case Western Reserve Univ., Cleveland, OH 44106, USA

Figure 1: My SOURCE Symposium and Poster Session research poster, which tied for 2nd place in the Engineering and Computer Science category.

This document discusses much of the theory and background needed to understand the project and goes into some depth about the LabView programs I wrote to conduct my experiments. It also presents the results of my preliminary experiments and the single-sensor experiment. A guide to using the software I have made will be delivered to Dr. White at a later date.

Background

The Robo-Moth Project

Figure 2: *Manduca sexta* moth from Dr. Mark Willis's lab. Photo courtesy of Z-Med Marketing Services.

This report represents a small portion of a larger project in which the odor-tracking abilities of moths are being explored. In Dr. Mark Willis's biology lab, North American tobacco hawkmoths (*Manduca sexta*) are studied for tracking speed and behaviors. Similarly, the Robo-Moth apparatus, in the Center for Biologically Inspired Robotics Research, is used as a platform for testing three-dimensional tracking algorithms using ionized air instead of pheromones (Rutkowski *et. al.*). Using data from both of these sources allows certain inferences as to how the moth—or robot—senses and tracks odor to be made. To fully understand what is happening, however, it is necessary to consider the environment in which the moth is tracking the odor. In other words, not only is it important to know how the moth moves toward an odor source, it is vital to understand how the odor moves away from its source.

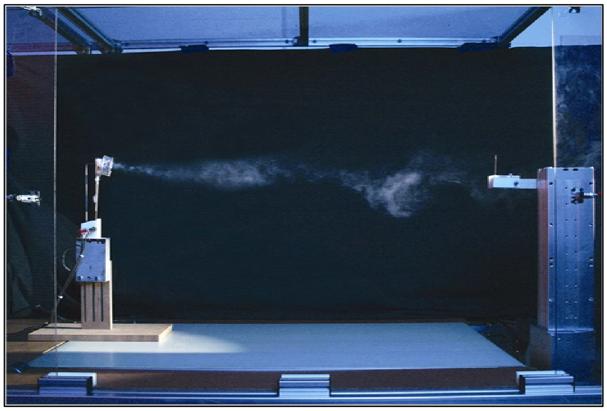


Figure 3: A turbulent chemical plume in the Robo-Moth wind tunnel. Photo courtesy of Z-Med Marketing Services.

When air moves over a stationary odor source, the air becomes seeded with molecules of odor and these molecules spread downstream in a roughly conical shape with the original source serving as the point of the cone. The resulting structure, known as an odor plume, is typically turbulent, meaning that its structure is in constant flux. Despite these continuous changes, plumes contain recognizable substructures known as eddies. Turbulent eddies are commonly found in nature; large-scale examples include clouds or the curls of smoke escaping a factory smoke stack. Although turbulent eddies are constantly being broken down into smaller and smaller substructures, new eddies are always being generated as well; therefore, large eddies can be found throughout the flow.

Of interest to us is the relative size of these largest eddies to moths—specifically to the distance between the moth's antennae. Because the eddy represents a single structure of a substance—in this case, an odor—one can assume that any measurement of a property of the substance will be closely related to a measurement of that same property elsewhere in the eddy. For example, if one were to measure the odor concentration at one point in the eddy, it is expected that the odor concentration at a second point in that eddy would be closely related to the first. This is because the eddy is essentially a clump of one material moving in a flow of another material (i.e. odor molecules moving in air). This can be applied to the moths thus: if an eddy is much larger than the distance between a moth's antennae, this indicates that both of the moth's antennae will usually be in the same eddy. The moth will, therefore, sense no significant difference in the odor between its antennae. If, on the other hand, a typical eddy is smaller than the distance between a moth's antennae, then it is likely that a moth will encounter an eddy of odor with one antenna while not measuring a similar odor with the other antenna. In this case, the moth derives a spatial advantage by having two antennae—very similar, in fact, to how humans are able to judge depth by having two eyes. Measuring the size of these eddies was the primary purpose of this experiment.

Measuring Characteristic Eddy Lengths

There are many ways in which one might measure the lengths of these eddies. One method is to use smoke or another chemical substance that can be seen to visualize the eddies. Pictures of the flow can then be analyzed and eddy sizes can be measured. This method was impractical for this project, though, in part because we did not have the equipment necessary for such flow visualization. The Robo-Moth facility, which uses ionized air to imitate an odor, on the other hand, already existed. Ions, though they cannot be seen by the human eye, can easily be measured: in essence, the ions represent a current flowing through the air and a wire is used to close the circuit and measure the amount of current the air is carrying. In this case, measurements of ion concentration need to be related to one another mathematically, so that the size of an eddy can be determined. As long as two measurements remain mathematically related, they are likely to belong to the same eddy.

This mathematical relationship takes the form of a statistical correlation:

$$C(\tau)=\frac{\frac{1}{T}\int_0^T f(t)\,f(t-\tau)\,dt}{\sqrt{\left(\frac{1}{T}\int_0^T\left[f(t)\right]^2 dt\right)\left(\frac{1}{T}\int_0^T\left[f(t-\tau)\right]^2 dt\right)}} \tag{1}$$

In this case, ***f(t)*** and ***f(t-τ)*** are fluctuating signals and ***C*** expresses the relation between them. If ***f(t)*** and ***f(t-τ)*** are identical, then ***C*** is equal to one. If they are entirely random and unrelated, ***C*** will be zero. For signals that are related but not identical, the magnitude of ***C*** will be between zero and one. It is also possibly for ***f(t)*** and ***f(t-τ)*** to be negatively correlated; if this is the case, then ***f(t)*** tends to be positive when ***f(t-τ)*** is negative and vice versa.

There are two basic types of correlation, the temporal correlation (autocorrelation) and the spatial correlation (cross-correlation). Both can be used to measure eddy lengths in different directions. In the case of autocorrelation, a single sensor can be used to measure the ion concentration at a single point over an extended period of time. The signal can then be correlated with itself by time-shifting portions of the signal and correlating the resulting signals. For example, a ten-second waveform signal can be broken into two, smaller signals, one which contains the signal from ***t*** = 0 to ***t*** = 8 seconds and one from ***t*** = 2 to ***t*** = 10 seconds. In this case the time-shift, τ, is equal to 2 seconds. The two signals can then be compared using Equation 1 and the resulting correlation coefficient, ***C***, will indicate how closely related the signal at a point ***t*** is to the signal two seconds later. Altering τ allows one to determine at which separation in time the signal becomes unrelated to what came before it. When the signal is no longer related to its time-shifted self, those two measurements represent the time it takes for an eddy to pass the sensor. Knowing the average speed with which the flow is moving, a length scale for the passing eddy can be defined.

A cross-correlation works very similarly except that two sensors are used instead of one. Rather than altering the time between signals, τ in a cross-correlation represents the physical distance between the sensors. When the physical separation becomes such that the two signals are uncorrelated, this indicates the physical size of an eddy passing between the two sensors.

Because eddy sizes can vary throughout the flow, it is important to take many measurements and perform many correlations to ensure that the length scale obtained is a reasonable average for the flow. For this reason, among others, it was vital that any experiments be automated. The nature and details of this automation is discussed later.

Methods

Experimental Set-Up

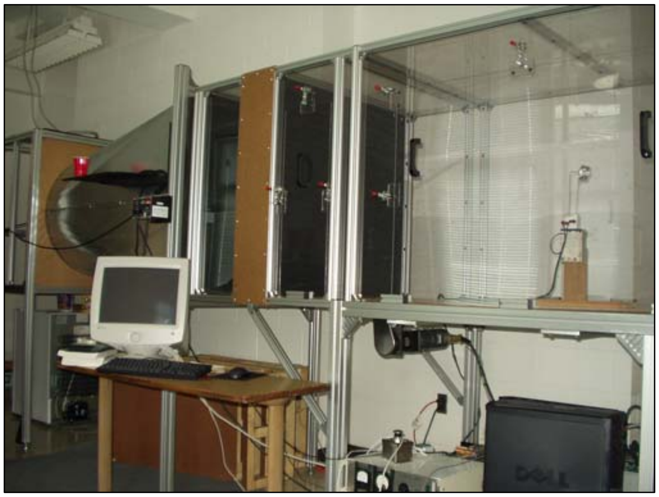


Figure 4: The Robo-Moth wind tunnel.

The Robo-Moth system's hardware can be broken into three basic categories: generation, motion control, and data acquisition. The first category, generation, contains the wind tunnel itself—an open pusher-fan configuration with a 1 m^2 test section—and the ion generator[1], which is driven by a DC power supply with a maximum potential of 4.1kV. At the end of the open test section is a 2-axis robotic gantry from the Adept 97N1-005 series. These allow movement of Robo-Moth's sensors in a spanwise cross-section of the wind tunnel test section. The gantry's motors are controlled by a Galil motion controller. They are connected to an ICM/AMP-1900 interconnect module, which, in turn, is controlled by a Galil DMC-1802 motion control card installed in a Dell computer with a 3GHz Intel Pentium 4 processor. Further detail on the wind tunnel, ion generator, and motion control hardware can be found in Rutkowski *et. al.* Data acquisition in the system is handled through a National Instruments NI cRIO-9215 4-channel, 16-bit module connected to the same Dell computer.

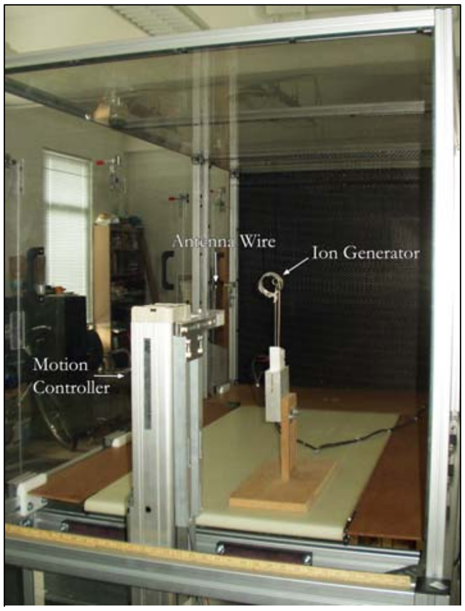


Figure 5: The test section of the Robo-Moth wind tunnel.

Automation of each experiment's motion control, data acquisition, and data analysis was done using custom software written in LabView 7.1, although some additional data analysis and graph generation was done using scripts in Matlab 7.1. The details of each experiment's custom software will be discussed later.

[1] Note that, although the photos of the Robo-Moth set-up include a foil shroud about the ion generator that this shroud was not present when the experiments reported were run.

Finally, a note on coordinate systems: for the purposes of this report, the x-axis is defined as the axis along which the horizontal motion controller moves (i.e. spanwise); the y-axis is the direction in which the vertical motion controller moves; and the z-axis is the direction of the flow through the test-section. This coordinate system is displayed in Figure 6.

Preliminary Experiments

Before the single-sensor turbulence experiments could be run, several aspects of the wind tunnel's operation needed to be characterized. First of all, it was important to verify that no significant net charge was building up as the wind tunnel ran with the ion generator on. In order to test this, I sampled ion concentrations in the wind tunnel for approximately thirty seconds at a time at 500 Hz, turning the ion generator on or off between each run. The mean and standard deviation for each run was saved and later plotted in Microsoft Excel.

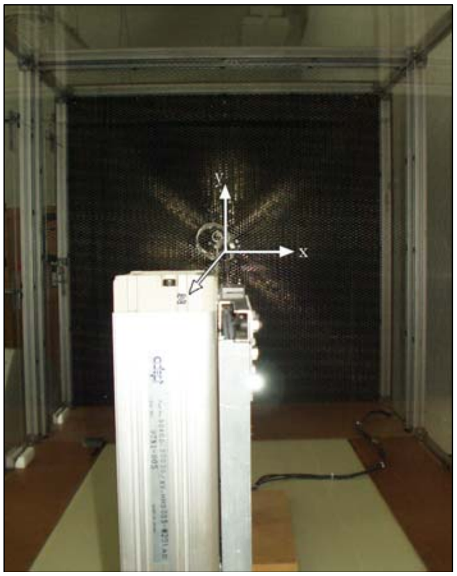


Figure 6: The coordinate system used.

Secondly, I characterized the wind tunnel's air speed in terms of fan speed. Rather than having an air speed controller, the Robo-Moth wind tunnel uses a Dayton DC Speed Controller to control fan speed and direction of rotation. In order to know approximately what speed at which the tunnel was operating, I borrowed a handheld, low-speed hotwire anemometer from Dr. Mark Willis and used it to characterize air speed according to the dial on the speed controller. Using a tripod stand to hold the anemometer, I recorded values from the device's LED screen over the course of several minutes before entering them into Microsoft Excel where the mean and standard deviation for each fan speed was calculated and plotted. These data were used later in calculating an eddy length scale from the time scales measured in the flow.

Single-Sensor Turbulence Statistics

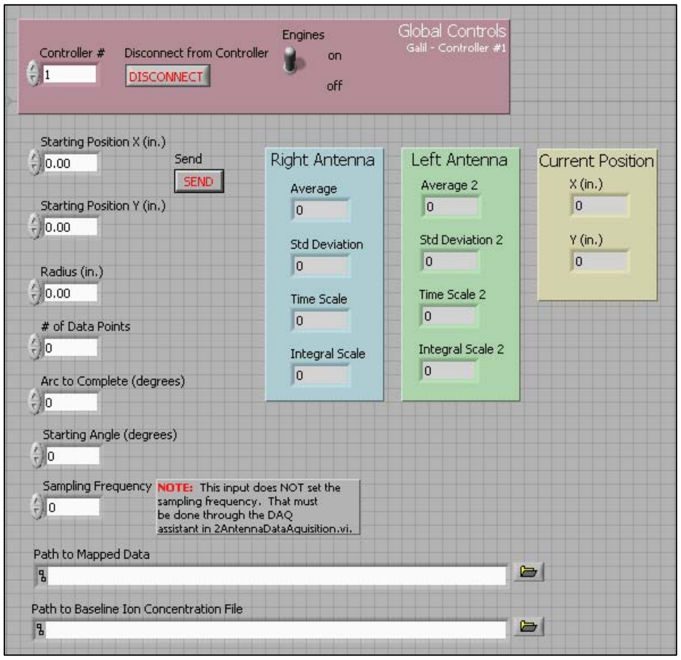
Data from a single sensor in the flow, as described earlier, can be used to find the characteristic eddy size in the Z-direction (i.e. streamwise direction) as well as to map characteristics of the plume. In order to do this as quickly and as accurately as possible, a LabView program was written to automate the experiment. The LabView program consisted of four segments: zero-ion testing, motion control, data acquisition, and data analysis. Each portion is discussed below.

Zero-Ion Testing

Because the preliminary experiments exploring the possibility of rising ion concentrations in the wind tunnel over time were inconclusive, the single-sensor turbulence testing procedure began by measuring and recording the ion concentration in the wind tunnel while the ion generator was off. Although this feature had not been integrated into the overall mapping program when the experiment was run, it has since been added so that, upon beginning an experiment, the user receives a message reminding him/her to turn off the ion generator while the zero-ion concentration is measured. Once this data has been taken, the user is told to turn the ion generator back on and the remainder of the experiment executes.

Motion Control

The Galil motion controller used to move Robo-Moth's sensors has its own BASIC-like language and an interface, DMC SmartView, that is used to control it. In order to automate motion control from LabView, some basic LabView VIs that allow communication with the controller itself were obtained from the manufacturer's website. Using these VIs as sub-programs within the mapping program allowed me to construct orders for movement into command strings that the motion controller recognizes while ensuring that a user need not know anything about this language to move the sensors in a meaningful way.



Controller # 1

Starting Position X (in.) 0.00

Starting Position Y (in.) 0.00

Radius (in.) 0.00

# of Data Points 0

Arc to Complete (degrees) 0

Starting Angle (degrees) 0

Sampling Frequency 0

NOTE: This input does NOT set the sampling frequency. That must be done through the DAQ assistant in 2AntennaDataAquisition.vi.

Path to Mapped Data

Path to Baseline Ion Concentration File

Right Antenna

Average 0

Std Deviation 0

Time Scale 0

Integral Scale 0

Left Antenna

Average 2 0

Std Deviation 2 0

Time Scale 2 0

Integral Scale 2 0

Current Position

X (in.) 0

Y (in.) 0

Figure 7: Front panel of the *Circular Movement.vi* program used to run experiments.

For the single-sensor statistics presented here, circular movement was used; however, a separate program that allows motion control along a line has also been made. In the first program, a user can specify a circular arc, its radius, starting angle, and number of points. The program will then calculate and construct the commands necessary to break this larger movement into appropriate step sizes. For example, in one of my runs, I specified a circle with a radius of 3 inches about the wind tunnel centerline to be sampled in 72 points. The program then broke this down such that the sensor was moved 5 degrees between each measurement. A picture of the LabView program, as seen by the user, is shown in Figure 7.

Data Acquisition

Data acquisition in LabView 7.0+ is somewhat different than in previous programs. Rather than being able to specify channels and frequencies for sampling directly in the front panel, it is necessary to go “behind-the-scenes” to change these values. In my program, these are part of the polymorphic VI called DAQ Task Assistant. Double-clicking on it in the block diagram allows a user to change inputs to the system such as the channels, frequency, and number of samples.

The program is designed for a number of samples that follows the following formula:

$$N = nf + 5 \tag{2}$$

where ***N*** is the number of samples, ***n*** is the number of whole seconds during which data is collected, and ***f*** is the sampling frequency, which should be a power of two. The reason for this complication in the number of samples has to do with the way data analysis is handled in the program.

In my experiments, I sampled from both the left and the right antenna for eight seconds at 512 Hertz. This particular frequency and length of time were chosen based on preliminary results which suggested that eddies passed through a point approximately every half a second and that these eddies required 0.3-0.4 seconds to resolve. By sampling for eight seconds, ten or more eddies would be passing the antennae, thus providing both better time-averaged maps of the plume and greater accuracy in judging characteristic eddy size.

Data Analysis

Once the program obtains ion concentration data from both antennae in the form of waveforms, the data analysis portion of the program takes over. Because a single experiment consisted of 20 runs of 72 points x 2 antennae—in other words, 2880 waveforms—some data analysis within the LabView program was needed to compress the experimental data into a reasonable size before saving it.

To do this, several values of interest were chosen. Firstly, the x- and y-locations recorded by the motion controller were saved. Secondly, the means and standard deviations of each antenna's waveform were required for mapping purposes. And, finally, some measure of the time scale[2] of the eddies passing through each antenna needed to be saved.

Before any data was logged, the data was passed through a digital low-pass filter (3rd order, Butterworth filter, cut-off frequency = 50 Hz). The mean of the data was then calculated and subtracted from every value, so that only the ion concentration fluctuations remained. The means and standard deviations of these data sets were then recorded.

Two methods were considered for calculating the time scale required for an average large eddy to pass through a single sensor. The first is based on the signal's energy spectra while the second is based on the concept of the integral scale.

An energy spectrum shows the relative importance of different frequencies to the overall measured signal. It was reasoned that the frequency with the highest energy would represent the largest eddies in the flow. Therefore, that frequency could be transformed back into the time domain and used as a time scale for the flow. The energy spectrum of a waveform was found by taking the Fourier transform of a waveform's autocorrelation:

$$E(f) = F\{C(\tau)\} \tag{3}$$

In Equation 4, also sometimes known as the Wiener-Khinchen theorem, ***E*** represents the energy spectrum as a function of frequency, ***F*** is a Fourier transform, and ***C(τ)*** is an autocorrelation function as described in Equation 1 (Tennekes and Lumley 210). The maximum value in ***E*** was then found and its corresponding frequency was transformed back into time and recorded.

The second method for estimating length was based on the integral scale, which is defined as:

$$T = \int C(\tau) d\tau \tag{4}$$

[2] The program itself calculates a time scale based on the data and saves it. Using the velocities found in the fan speed characterization, changing this time scale into a length scale is trivial.

where *C(τ)* again is the autocorrelation of a signal. The integral scale represents "a rough measure of the interval over which [a signal] is correlated with itself" (Tennekes and Lumley 211). This value was also calculated based on each waveform's autocorrelation before being recorded.

Recall that Equation 2, which described the number of samples taken, required the frequency at which a sample was taken to be a power of two. This is because LabView uses a fast Fourier transform (FFT) when taking an autocorrelation, and FFTs work best when conducted over intervals of a power of two.

To further increase the accuracy of both the energy-spectra-based time scale and the integral scale, the data-windowing method described in Chapter 13.4 of *Numerical Recipes in C* was used. The full eight second sample was broken down into seven, overlapping two-second intervals (***t*** = 0 to ***t*** =2 seconds, ***t*** = 1 to ***t*** = 3 seconds, ***t*** = 2 to ***t*** = 4 seconds, etc.). The energy spectra and integral scales of these smaller intervals were then calculated and recombined into an average value, which was saved for further analysis.

To recap, the following values were saved by the LabView program:

- X-location from the motion controller
- Y-location from the motion controller
- Means of the ion fluctuations for the left and right antenna
- Standard deviations of the ion fluctuations for the left and right antenna
- Energy-spectra-based time scale for the left and right antenna
- Integral-scale-based time scale for the left and right antenna

A total, therefore, of ten numbers was saved for each point in the flow that was tested. Since the experiment was completed, I have, by request, added the unfiltered means and standard deviations of the ion fluctuations of both antennae as further values to be saved in the future. Additional screenshots of the program can be found in Appendix A.

For the experiment, I did my best to align the left antenna so that it was on the centerline of the wind tunnel. The right antenna was one inch to the right of it. The ion generator was also located on the centerline of the wind tunnel at a distance of one meter downstream of the antennae. The experimental procedure outlined above was then completed and data was recorded for each of the plume locations shown in Figure 8.

Once the experiment was complete, the data saved by LabView was transferred into Matlab 7.1 for further data analysis. The data were first adjusted by subtracting the zero-ion concentrations from the means and standard deviations. Then the x-location of the data from the right antenna was moved one inch further to the right before the data from both antennae were combined. The combined data were then used to create surface and contour plots. In cases where measurements from both the left and right antenna existed at a single point, an averaged value was used. Any time scale values were also multiplied by the air speed to adjust them to length scales. These results are reported below. Portions of the Matlab code used can be found in Appendix B.

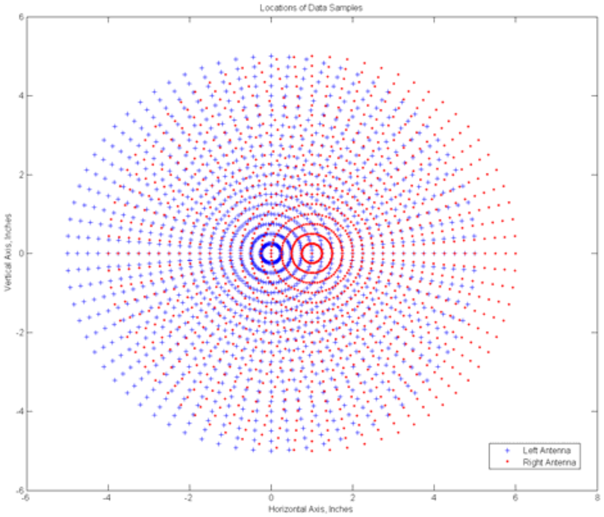


Figure 8: The locations at which data was sampled. Blue represents data from the left antenna and red represents data from the right antenna.

Results

Zero-Ion Concentrations

Figure 9 shows one of the zero-ion concentration tests. In this case, the ion generator was turned on for approximately thirty seconds before being switched off while these data were collected. The zero-ion concentration was sampled for approximately thirty seconds at 500 Hz. The ion generator was then turned back on and the procedure was repeated. Individual points represent the mean ion concentration with the generator off, while the error bars display the standard deviation. Points 4 and 5 have much larger standard deviations than previous points; this is because they were taken with the wind tunnel on, while Points 1-3 were taken with the wind tunnel off.

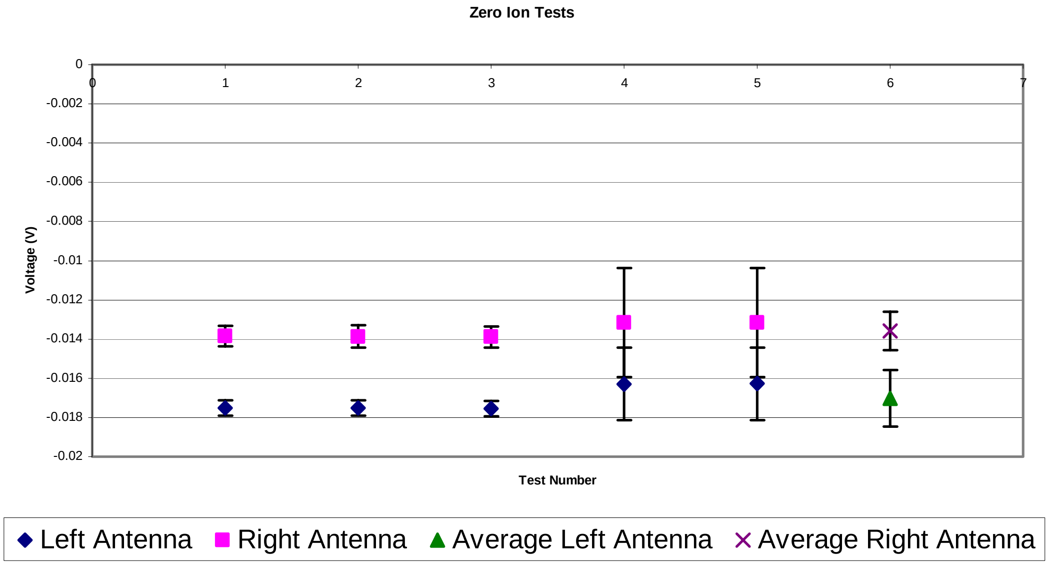


Figure 9: The mean ion concentration measured by each antenna while the ion generator was off immediately after a period when it had been on. Error bars display the standard deviation of the signal.

Because these data are not particularly conclusive and do not indicate that the ion concentration in the wind tunnel does not rise with time, the single-sensor turbulence experiments that followed were structured such that the mean ion concentration without the ion generator on was tested between each experimental run. In this way, it was possible to monitor any rise in the baseline ion concentration of the tunnel test section.

Characterization of Air Speed as a Function of Fan Speed

The results of the flow speed characterization can be found in Figure 10. Averages and standard deviations for each fan speed are listed in Table 1. Note the increasing unsteadiness of the air speed as the fan speed increases. Also worth noting is the slight increase in average speed as the tunnel runs longer than one or two minutes.

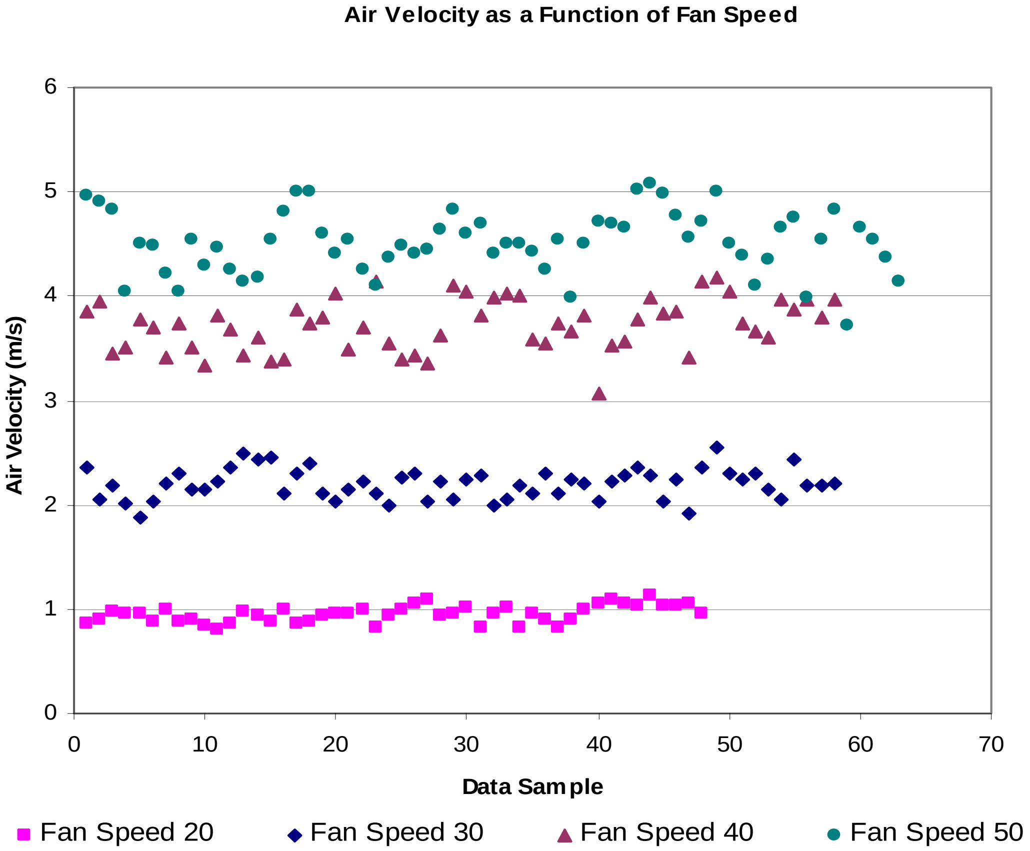


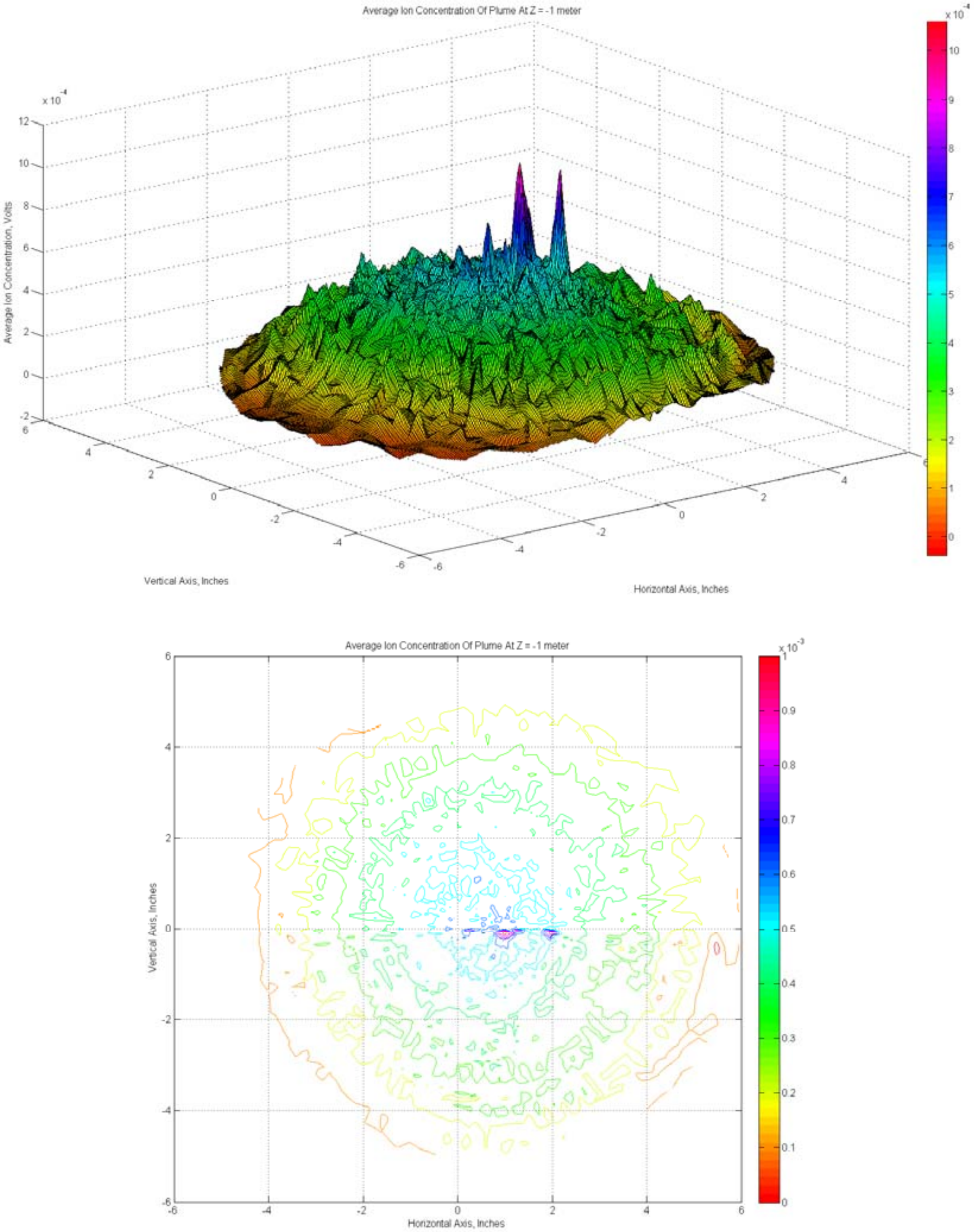
Figure 10: Air velocity as a function of fan speed in the Robo-Moth wind tunnel.

Table 1: Air Speed as a Function of Fan Speed

Fan Speed	Mean (m/s)	Standard Deviation (m/s)
20	0.951	0.080
30	2.200	0.146
40	3.723	0.249
50	4.385	0.326

Single-Sensor Turbulence Statistics

Figure 11 shows a 3D surface plot of the mean concentration of the ion fluctuations across a cross-section of the plume located one meter downstream of the ion generator. Figure 12 shows the same data in the form of a contour map. The data in both graphs represent a conglomerate of the data from the right and left antennae and were adjusted for the zero-ion concentrations for each antenna before being combined in Matlab. When the data were collected, the ion generator was located along the wind tunnel's centerline at (0,0). Note, however, that the peak concentrations are not located along the centerline. There are several possible explanations for this behavior, and they will each be discussed later.



Figures 11 and 12: Mean concentration fluctuations across a cross-section of the ion plume located one meter downstream of the ion generator.

Additionally, both Figure 11 and Figure 12 show that the plume is essentially axisymmetric about its center point.

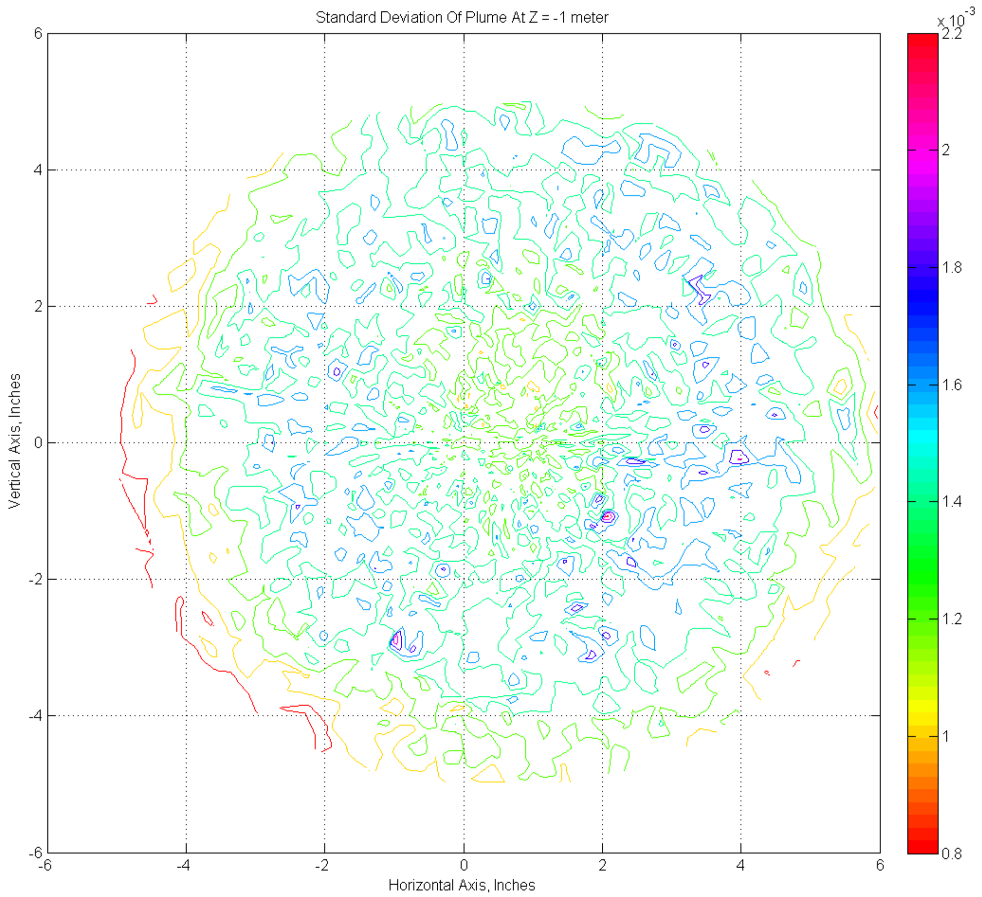


Figure 13: Contour map of the standard deviation of the ion fluctuations across a plume cross-section located one meter downstream of the ion generator. Note that this value is equivalent to the root mean square of the ion concentration.

Figure 13 shows that the standard deviation of the ion fluctuations is also axisymmetric about a point off-set from the wind tunnel centerline. However, unlike the mean of the concentration fluctuations, the standard deviation goes from low in the center to high and then back to low as one moves out radially. In three dimensions, this creates a shape similar to that of a Bundt cake. Figure 14 demonstrates this shape. Rather than showing a full three-dimensional plot, Figure 14 shows the standard deviation values along the $y = 0$ line.

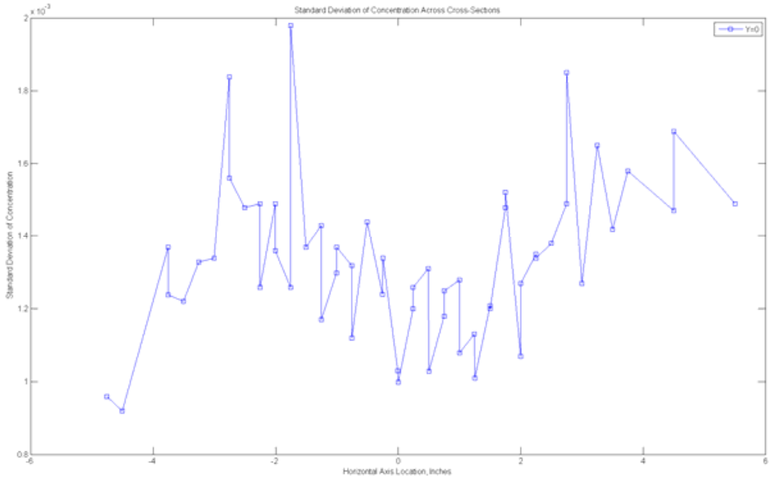


Figure 14: The standard deviation of the concentration fluctuations along the line $y = 0$ in a cross-section of the plume located one meter downstream of the ion generator.

Finally, Figure 15 shows the characteristic eddy length in the Z-direction across the same plume cross-section. This length scale is based upon the integral scale, as discussed in the Methods section. Interestingly, there is virtually no variation in this eddy length across the cross-section.

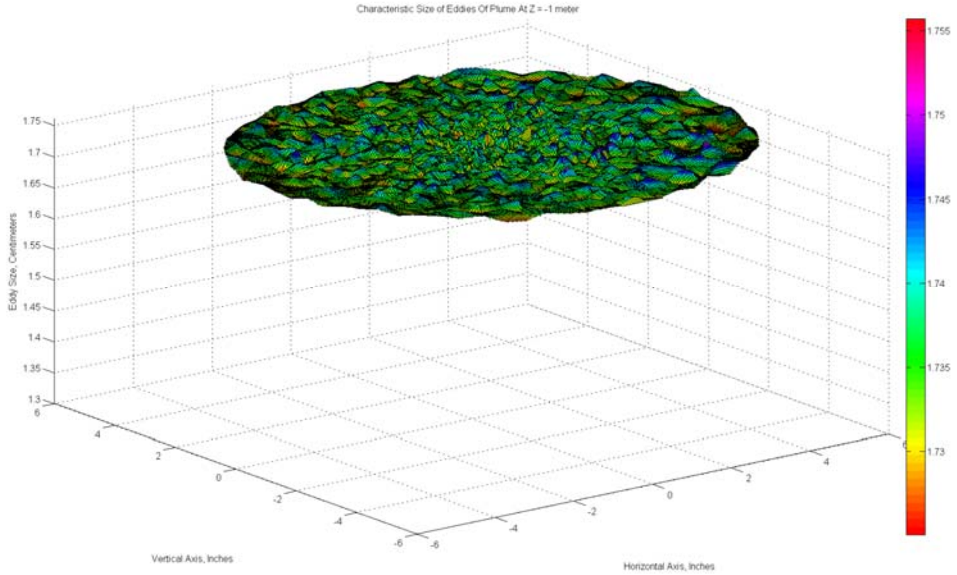


Figure 15: The characteristic eddy length in the Z-direction, as based on the integral scale, across a cross-section located one meter downstream of the ion generator.

Although the length scale based on the energy-spectrum was also calculated, the results are not shown. I suspect that an error exists in my implementation of the theory for this measurement because the results involve eddy lengths on the order of ten meters.

Discussion

As seen in Figures 11 and 12, the largest mean ion fluctuation concentrations are not located along the centerline of the wind tunnel as might be expected. One possible reason for this is a misalignment of the antennae. As noted in the Methods section, I tried to bend the left antenna such that it was aligned with the centerline of the wind tunnel; however, it is likely that there was an unknown offset. Because the right antenna was one inch away from the left antenna, it seems as though any offset would simply shift the entire map. I doubt, however, that this entirely accounts for the offset shown in the data because there is a vertical offset as well.

A second possibility is plume meander. In nature, plumes do not follow a straight line, and their center varies across different cross-sections. This is consistent with moth behavior as it tracks an odor. First, a moth turns into the wind, then it switchbacks back and forth as it flies upwind (Rutkowski *et. al.*). If the moth simply located the center of a cross-section and attempted to fly directly into the wind from there, it would lose the odor due to plume meander.

A final possibility is that there is some vorticity in the flow induced by the fan of the wind tunnel. As the vortical flow passes the ion generator, ions are picked up and carried along it. In this case, sampling varying cross-sections would reveal a helical shape of high ion concentrations along the streamwise axis. Such measurements could also be used to determine whether this is simply a case of plume meander.

Regardless, the mean and standard deviation maps provided here, along with the ability to automate similar experiments at different distances from the ion generator, represent a new source of data for the Robo-Moth project. By further characterizing the flow in the tunnel, they may lead to a better tracking algorithms and parameters for Robo-Moth while also indicating potential room for improvement in the tunnel itself—if, for example, vorticity is a problem in the tunnel.

The characteristic streamwise eddy size of approximately 1.73 cm is a start toward estimating the size of the eddies a moth encounters. To get the x- and y- eddy lengths, a double-sensor experiment using cross-correlations, as discussed earlier, is necessary. This experiment is in the process of being set-up. The automation program, which is closely based on the single-sensor program presented here, is nearly written, and the stationary sensor is in the process of being fabricated. I expect to have at least preliminary double-sensor data within the next week at which point it will be possible to compare a full eddy to the distance between a moth's antennae, perhaps even accounting for changes in orientation as a moth flies. If nothing else, I will complete the documentation and programming necessary for someone else to perform the experiment after I am gone.

Conclusions

This report detailed the automation of a single-sensor mapping program capable of measuring streamwise eddy lengths in a plume of ionized air in the Robo-Moth wind tunnel. The single-sensor experiment conducted one meter downstream of the ion source showed that the plume, though axisymmetric as expected, is offset from the centerline of the wind tunnel. Possible causes for this were discussed and a method for determining which cause is most likely was suggested. Additionally, the results showed a streamwise eddy length of 1.73 cm, which remains consistent across the plume.

The basics of a double-sensor experiment to determine the x- and y-eddy lengths were also presented. Once that experiment—or the documentation necessary for another to conduct that experiment—is complete, this report will be properly expanded and turned in to Dr. White.

Acknowledgements

I would like to express my thanks to Dr. Edward White, who has guided me throughout this project, and to Drs. Roger Quinn and Mark Willis, who allowed me to use their equipment. Special thanks are also due to Adam Rutkowski, without whom this experiment would never have been possible, and to everyone in the Center for Biologically Inspired Robotics Research.

This work was originally funded by a summer fellowship Support of Undergraduate Research and Creative Endeavors (SOURCE) program with money generously donated by the Case Alumni Association.

References

Press, William H. *et al.* *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1992.

Rutkowski, A. *et al.* A robotic platform for testing moth-inspired plume tracking strategies. From the IEEE International Conference on Robotics and Automation (ICRA '04), New Orleans, 2004. <http://biorobots.cwru.edu/publications/ICRA04_Rutkowski_RoboMoth.pdf>

Tennekes, H. and Lumley, J. L. *A First Course in Turbulence*. MIT Press, 1994.

Warhaft, Z. Passive scalars in turbulent flow. *Annu. Rev. Fluid Mech.*, 32:203-240, 2000.

Warhaft, Z. Turbulence in nature and in the laboratory. From the Arthur M. Sackler Colloquium of the National Academy of Sciences, "Self-Organized Complexity in the Physical, Biological, and Social Sciences," March 23-24, 2001.

Webster, D. R. and Weissburg, M. J. Chemosensory guidance cues in a turbulent odor plume. *Limnol. Oceanogr.* 46(5):1034-1047, 2001.

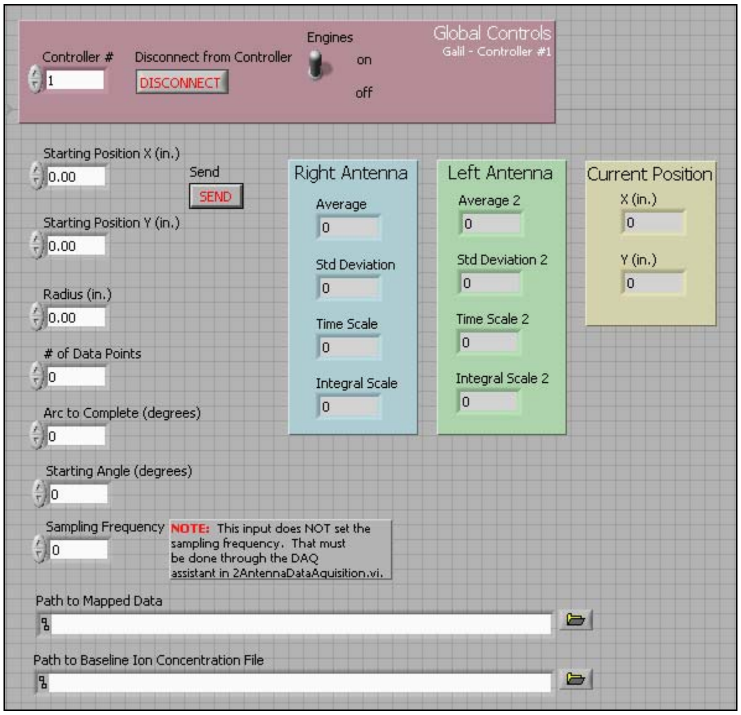
Webster, D. R., Rahman, S., and Dasi, L. P. On the usefulness of bilateral comparison to tracking turbulent chemical odor plumes. *Limnol. Oceanogr.* 46(5):1048-1053, 2001.

White, E. B., Willis, M. A., and Birch, E. D. Coherent turbulent structures and flight control of odor-tracking insects. From the Proc. 7th World Multi Conf. on Systems, Cybernetics, and Informatics, 2003.

Appendix A: LabView VI Screenshots

Circular Movement.vi

The following screenshots are from the LabView program intended for use with circular-type motion.



Controller # 1

Disconnect from Controller

Starting Position X (in.) 0.00

Starting Position Y (in.) 0.00

Radius (in.) 0.00

# of Data Points 0

Arc to Complete (degrees) 0

Starting Angle (degrees) 0

Sampling Frequency 0

NOTE: This input does NOT set the sampling frequency. That must be done through the DAQ assistant in 2AntennaDataAquisition.vi.

Path to Mapped Data

Path to Baseline Ion Concentration File

Right Antenna: Average 0; Std Deviation 0; Time Scale 0; Integral Scale 0

Left Antenna: Average 2 0; Std Deviation 2 0; Time Scale 2 0; Integral Scale 2 0

Current Position: X (in.) 0; Y (in.) 0

Figure 16: The front panel of the program for an experiment with circular motion.

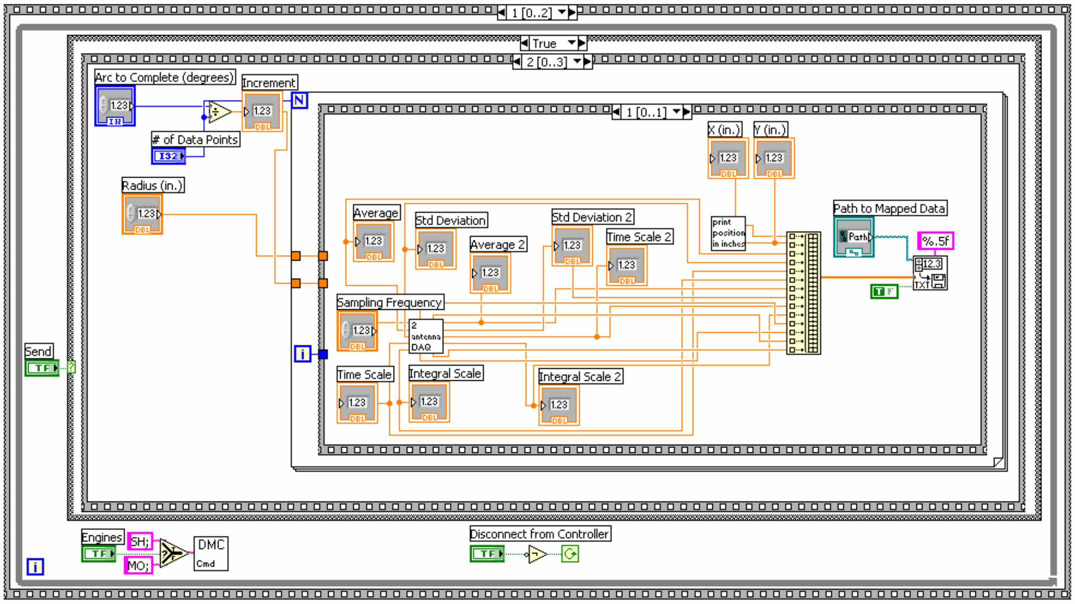
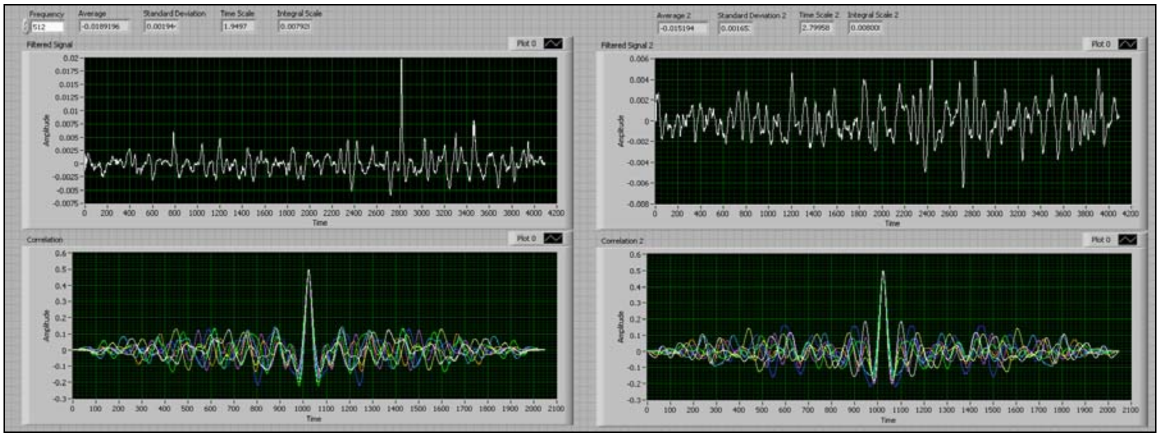


Figure 17: Part of the block diagram for the LabView program for an experiment using circular motion.



Frequency 512 | Average -0.0189196 | Standard Deviation 0.00194 | Time Scale 1.9497 | Integral Scale 0.00792

Average 2 -0.015194 | Standard Deviation 2 0.00165 | Time Scale 2 2.79958 | Integral Scale 2 0.00800

Figure 18: The front panel for the data acquisition sub-VI, showing the ion fluctuation signal (top) and autocorrelations (bottom) for each antenna.

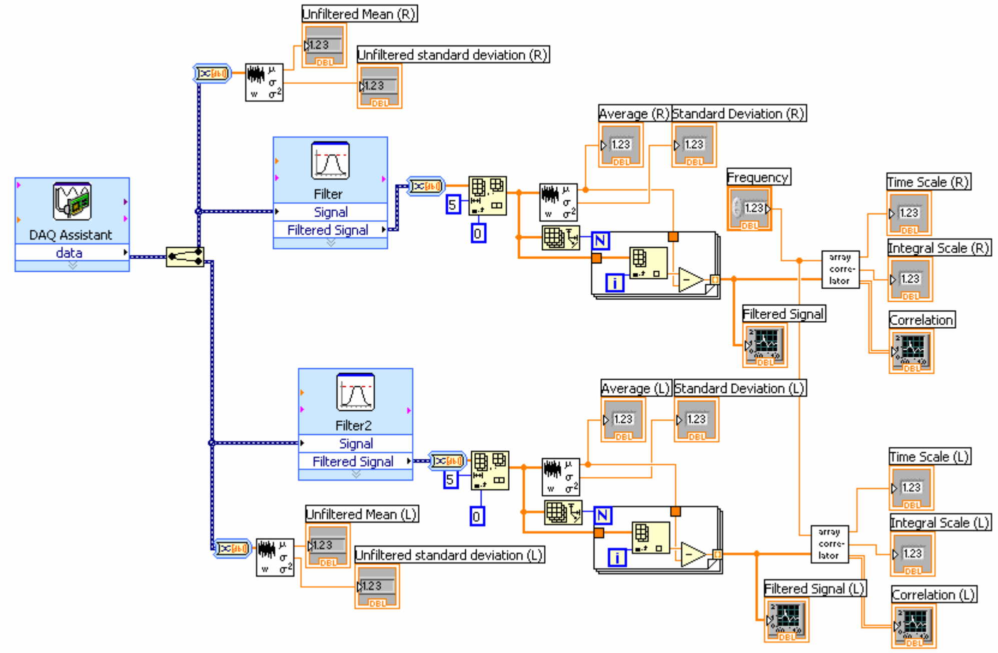


Figure 19: The block diagram for the data acquisition sub-VI shown above.

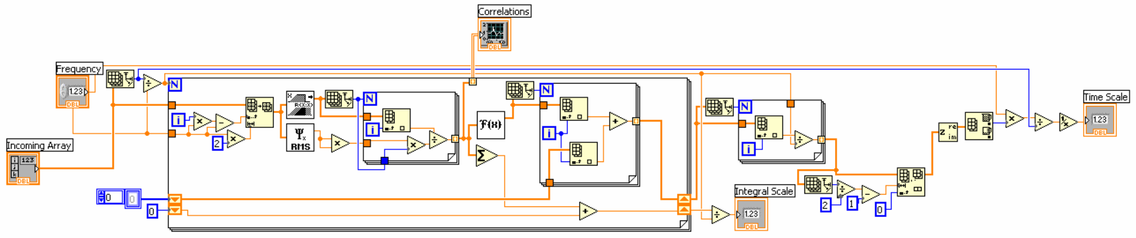
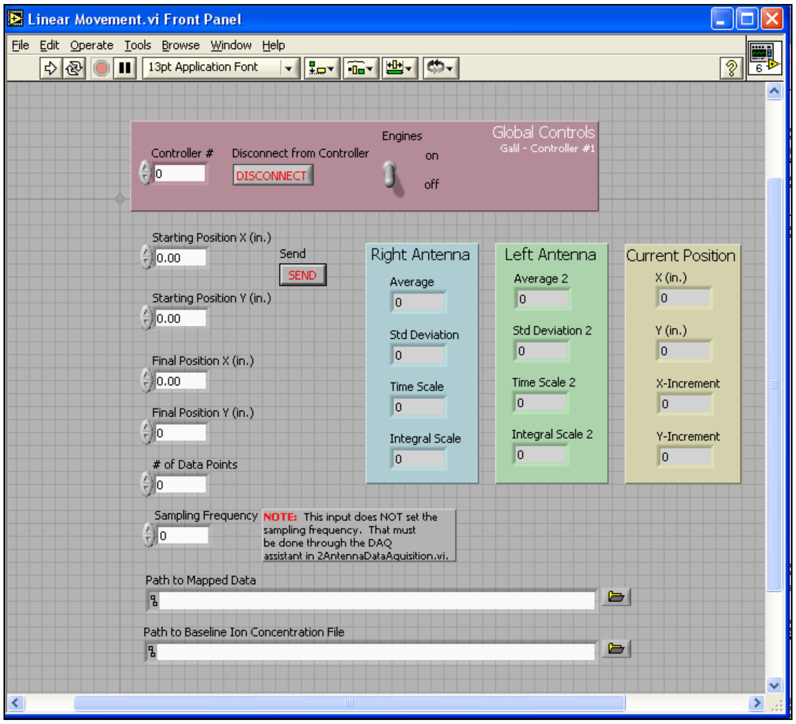


Figure 20: A sub-VI used in the data acquisition sub-VI. This program calculates the energy-spectra- and integral-scale-based time scales.

Linear Movement.vi

The following screenshots are from the LabView VI intended for use in experiments with linear motion.



Linear Movement.vi Front Panel

Global Controls
Galil - Controller #1

Controller # 0

Disconnect from Controller DISCONNECT

Engines on off

Starting Position X (in.) 0.00

Send SEND

Starting Position Y (in.) 0.00

Final Position X (in.) 0.00

Final Position Y (in.) 0

# of Data Points 0

Sampling Frequency 0

NOTE: This input does NOT set the sampling frequency. That must be done through the DAQ assistant in 2AntennaDataAquisition.vi.

Path to Mapped Data

Path to Baseline Ion Concentration File

Right Antenna

Average 0

Std Deviation 0

Time Scale 0

Integral Scale 0

Left Antenna

Average 2 0

Std Deviation 2 0

Time Scale 2 0

Integral Scale 2 0

Current Position

X (in.) 0

Y (in.) 0

X-Increment 0

Y-Increment 0

Figure 20: The front panel for the program for an experiment with linear motion.

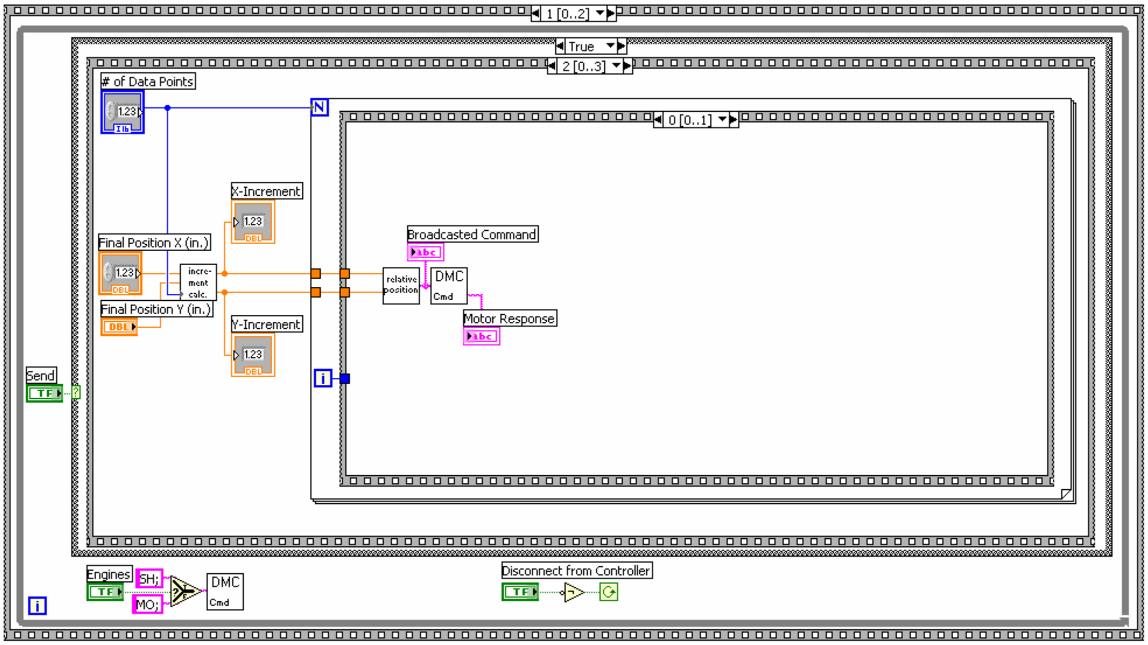


Figure 21: A screenshot of the block diagram for the program above, showing, among other things, the transmission of a command to the motion controller.

Appendix B: Matlab Code

This appendix contains some of the Matlab code written for additional analysis and for the display of some of the graphs shown in the Results section of this paper.

```
clear all;
zeroion = dlmread('zeroiontest', '\t');
s = length(zeroion);
for j=1:s,
    Rm(j,1)=zeroion(j,1);
    Lm(j,1)=zeroion(j,3);
    Rsd(j,1)=zeroion(j,2);
    Lsd(j,1)=zeroion(j,4);
end
Rmean=mean(Rm);
Lmean=mean(Lm);
Rstd=mean(Rsd);
Lstd=mean(Lsd);
waveforms = dlmread('map','\t');
L = length(waveforms);
n =1; %% Unit length conversion factor for changing from inches to whatever
for i=1:L,
    X(i,1)=n*waveforms(i,1);
    XL(i,1)=X(i,1);
    XR(i,1)=n*waveforms(i,1)+n;
    Y(i,1)=-n*waveforms(i,2);
    Y1(i,1)=Y(i,1);
    ZR(i,1)=-(waveforms(i,3)-Rmean); %% Correct for offset in right antenna
mean
    RSD(i,1)=waveforms(i,4)-Rstd; %% Right antenna standard deviation
    RTS(i,1)=2.2*waveforms(i,5); %% Right antenna length scale
    RIS(i,1)=220*waveforms(i,6); %% Right antenna integral length scale
    ZL(i,1)=-(waveforms(i,7)-Lmean); %% Correct for offset in left antenna
mean
    LSD(i,1)=waveforms(i,8)-Lstd; %% Left antenna standard deviation
    LTS(i,1)=2.2*waveforms(i,9); %% Left antenna length scale
    LIS(i,1)=220*waveforms(i,10); %% Left antenna integral length scale
end
for i=1:L,
    X(i+L,1)=XR(i,1);
    Y(i+L,1)=Y(i,1);
    ZL(i+L,1)=ZR(i,1);
    LTS(i+L,1)=RTS(i,1);
    LIS(i+L,1)=RIS(i,1);
    LSD(i+L,1)=RSD(i,1);
end
ti = -6:.04:6;
[XI,YI] = meshgrid(ti,ti);
ZI = griddata(X,Y,LIS,XI,YI);

%{
H = length(X);  Cut out and graph cross-sections of the st. dev. data
j=1;
q=1;
for i=1:H,
    if Y(i,1) == 0,
        A(j,1)=i;
        j=j+1;
    else  if (X(i,1) <= 0.001) & (X(i,1) >= -0.001),
            B(q,1)=i;
            q=q+1;
        else continue;
        end
    end
end
a = length(A);
b = length(B);
for i=1:a,
    k = A(i,1);
    X_sd(i,1)=X(k,1);
    LSD_sd(i,1)=LSD(k,1);
end
for i=1:b,
    f = B(i,1);
    Y_sd(i,1)=Y(f,1);
    LSD_ysd(i,1)=LSD(f,1);
end
G = horzcat(X_sd,LSD_sd);
F= sortrows(G);
g = length(F);
for i=1:g,
    x_sd(i,1)=F(i,1);
    L_sd(i,1)=F(i,2);
end
U = horzcat(Y_sd,LSD_ysd);
V = sortrows(U);
v = length(V);
for i=1:v,
    y_sd(i,1)=V(i,1);
    L_ysd(i,1)=V(i,2);
end
figure
%plot(x_sd,L_sd,'-sb',y_sd,L_ysd,'-dr');
plot(x_sd,L_sd,'-sb');
title('Standard Deviation of Concentration Across Cross-Sections');
xlabel('Horizontal Axis Location, Inches');
ylabel('Standard Deviation of Concentration');
legend('Y=0','X=0');
%}

%LISmean = mean(LIS)
figure
contour(XI,YI,ZI);
colormap hsv
title('Characteristic Size of Eddies Of Plume At Z = -1 meter');
xlabel('Horizontal Axis, Inches');
ylabel('Vertical Axis, Inches');
zlabel('Eddy Size, Centimeters');
hidden on;
grid on;
```

```
%{
% Show locations of data collection points
figure
plot(XL,Y1,'+b',XR,Y1,'.r');
legend('Left Antenna', 'Right Antenna');
title('Locations of Data Samples');
xlabel('Horizontal Axis, Inches');
ylabel('Vertical Axis, Inches');
%}
```