2D Turbulence Experiments: Overview

Michael Schatz, Balachandra Suri, and Jeffrey Tithof (Georgia Tech)
Xiang Wu (Shanghai Jiao Tong University)

Purpose

This experiment provides an opportunity to test recent theoretical advances in turbulence, one of today's great unsolved problems in physics. Specifically, the Hands-On School experiments examine quasi-2D flows for evidence of unstable periodic solutions, which recent theories suggest may be used to characterize turbulent flows. The experiments are original research and may lead to publishable results that help advance the understanding of turbulence.

The main elements covered in this document are:

1. Theoretical Background
2. Experimental Setup
3. Flow Visualization and Image Acquisition
4. Image Data Analysis

This document gives a very brief overview of these elements; further information and background will be found both in weblinks referred herein and files that accompany this document.

1. Theoretical Background

Turbulence has long been associated with the idea of fluid flow that is irregular and unpredictable. However, decades of experimental observations demonstrate that characteristic patterns arise repeatedly. Numerous empirical methods have been devised to characterize these patterns, known as "coherent structures"; however, recent theory suggests that coherent structures observed in experiments are closely related to certain unstable solutions in the fluid equations (the Navier-Stokes equations). Firm experimental evidence supporting the connection between these unstable solutions and coherent structures has not yet been found.

The theory to date has focused on flows in 3D (pipe flow, plane Couette flow); clear experimental evidence requires measurement of 3D time-dependent velocity fields, which is at the limit of today's experimental capabilities. By contrast, measurement of 2D velocity fields in fluid flows is relatively easy. At present, there is no theory for unstable solutions in 2D flows; results from your experiments may help stimulate future theory.

Further reading:

A general overview of research on unstable solutions in plane Couette flow can be found at: http://chaosbook.org/tutorials/

The folder "theory_background" contains scientific journal articles that describe recent work (mostly theory) on unstable solutions and turbulence.

2. Experimental Setup

The experiment uses electromagnetic forces to drive flow in a shallow layer of electrolyte. Permanent magnets with high field strength are placed in/under a plastic box; the box is then filled with a thin layer of electrolyte (copper sulfate). A current is passed through the electrolyte (green arrows), causing Lorentz forces to drive the flow (blue and yellow arrows). For small currents, the flow reflects symmetries (if any) of the magnets' spatial arrangement; however, for sufficiently large currents, the flow breaks these symmetries and appears turbulent. Previous work has suggested the flow is quasi- 2D (i.e., the in-plane velocity fields are different at different vertical positions; however, these in-plane fields are weakly coupled to one another due to the smallness of the vertical velocities in the layer.) A number of earlier experiments in this system have explored the statistics of 2D turbulence (e.g., energy and enstrophy cascades) and, more recently, the problem of turbulent mixing in 2D. However, no previous experiments have explored the role of unstable solutions in 2D turbulence.

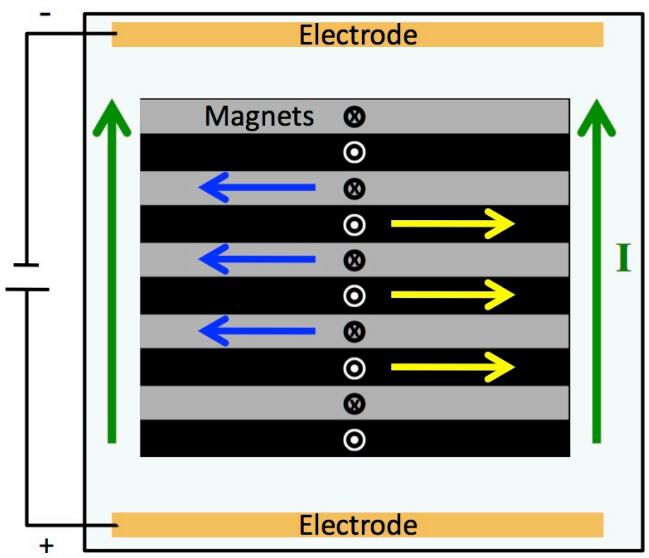

Figure 1: View from above of the experimental setup for Kolmogorov-like flow. Current (green arrows) is passed through an electrolyte; in the region above several permanent magnets, Lorentz forces (yellow and blue arrows) drive shear flow. As the current is increased, the system transitions into turbulence.

A note about power supplies to run the experiment: Any readily available power supply with a range of approximately 5V and 0.3 A should be sufficient for the type of experiment apparatus similar to the one you are working with in this lab. Ideally, the supply should be operated in current control mode; this fixes the electromagnetic forcing (which is proportional to the applied current). Here, we use an inexpensive current-controlled supply built from a cell-phone charger and a few electronic parts (for a detailed discussion on how to construct a suitable current-controlled power supply, see "power_supply.pdf", which accompanies this document.)

Further reading:

The folder "exp_background" contains selected scientific journal articles (including review articles) that describe earlier table-top experiments on 2D turbulence, including experiments on electromagnetically-driven electrolytes.

3. Flow Visualization and Image Acquisition

The flow can be observed by sprinkling glass microspheres (mean diameter—40 micrometers) on the electrolyte layer. The glass microspheres float at the layer surface; thus, when the fluid layer is in motion, the glass microspheres "track" the flow at the top of the fluid layer. (The microspheres must be sufficiently small to respond rapidly.)

A web camera that connects to the computer is used to image the visualized flow. We use the Fire-I web camera, whose software drivers (freely downloadable from the web) permit setting both the frame rate and shutter speed. Any web camera can be used to obtain suitable images; for general web cameras, we suggest using the freeware application, VirtualDub, for image acquisition. The VirtualDub executable and documentation are easily found by a keyword search ("VirtualDub") online. Regardless of the hardware/software used for the acquisition, the images need to be stored in .tif format in preparation for processing to find fluid velocities.

Further reading:

For more information on image acquisition, feel free to contact: mike.schatz@physics.gatech.edu

4. Image Data Analysis

4.1 Summary

After a time series of suitably named images (e.g., 'Frame_0001.tif') has been collected, the image data can be processed. The following is the sequence of Matlab scripts to execute to process a time series of images.

Matlab Scripts (Listed in Execution Order)	Purpose
image_process('startnum', 'stopnum', h_cm, fps);	Calls OSIV software to perform PIV.
taverage;	Time averages data to reduce noise.
vfield_cgs;	Converts velocities and lengths to cgs units.
vortex_cgs;	Calculates vorticity contour in cgs units.
remove_vortex_noise;	Removes noisy parts of vorticity to make data look smoother.
make_video(jump, video_fps);	Generates an AVI video of the flow.
recurrence;	Generates a recurrence plot of the velocity field time-series.

The arguments for *image_process* are:

'startnum' – a string containing the number portion of the first input image to be processed, with leading zeros (if applicable)

'stopnum' – a string containing the number portion of the last input image to be processed, with leading zeros (if applicable)

h_cm – the height of the image in centimeters; can be obtained by imaging a ruler

fps – the rate at which the data was collected (in frames per second)

Example: image_process('002', '900', 4.8, 7.5)

The arguments for *make_video* are:

jump – the number of time steps to jump over between consecutive frames of the video; a larger number makes the video speed up (8 is suggested)

video_fps – the speed at which to encode the movie in frames/second (15 is suggested)

Example: make_video(8, 15)

4.2 Details

There are several steps to image analysis that are performed by the numerous scripts included in the "image_analysis" folder. The first and most important step is particle image velocimetry (PIV). PIV is a process in which a time-series of a velocity field can be extracted from images of a fluid flow taken in rapid succession. To perform the PIV analysis, we use the open source package OSIV. This package can be installed under Windows, Mac OS X, or Linux and run as a toolbox from within the MATLAB environment. (The package can also be installed and run directly from the command line in these operating systems.) Once OSIV is installed, our scripts can be executed.

For simplicity, one only needs to call the function *image_process* to perform PIV. To execute *image_process*, while in the directory containing a time-series of images, type:

image_process('startnum', 'stopnum', h_cm, fps)

where "startnum" and "stopnum" are the indices corresponding to the first and last image to be processed, h_cm is the height of the image in centimeters, and fps is the rate at which the data was collected (in frames per second). The h_cm input is used to scale pixel units to centimeters, and we usually obtain h_cm by taking a picture of a carefully positioned ruler (see **Figure 2**). Note that the first two input arguments are strings, so if the first file is Frame_002.tif, the last file is Frame_900.tif, a picture of a ruler shows that the height of our image is 4.8 cm, and the data was collected at 7.5 frames per second, then one would type:

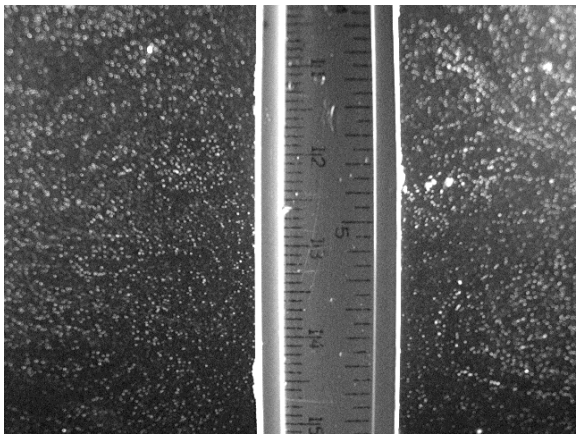

Figure 2: By imaging a ruler at the level of the fluid flow, one can determine the value of "h_cm" to use as an argument for *image_process.* Here, h_cm is 4.8.

image_process('002', '900', 4.8, 7.5)

where the zeros are necessary to include. The program will warn you if a frame is missing, as will occasionally happen when the camera skips a frame. For a more thorough discussion of the methods and output of our scripts, continue reading below. Explanation of the scripts are available in the files, themselves, which are well commented.

Particle Image Velocimetry (PIV): This technique is a widely-used method of measuring velocity fields. To determine a single velocity field "snapshot" requires two images of the tracer particles in the flow; the images are recorded at slightly different times, such that the tracer particles have moved a little bit (but not too much) in the time interval between the two images. Each image is then divided into a collection of interrogation windows. A spatial cross-correlation is performed between the pair of corresponding windows; the shift that produces the maximum in the cross-correlation corresponds to the most likely displacement vector associated with the windows; by knowing the time interval between the images, we can find an estimate of the average velocity vector (displacement vector / time interval) associated with the pair of windows. Repeating this process for all interrogation windows yields a set of velocity vectors distributed on a uniform

grid, i.e., the velocity field snapshot for the two images. By performing this same analysis between each pair of adjacent (in time) images from a time series of the visualized flow, we can determine the corresponding flow velocity field time series. Our function *pivCorrelate_fi(first,last)* is what sets the PIV specifications and calls the OSIV software.

AVI Video: A video file of the flow, sped up to several times real time, is a convenient and useful way to visualize data.

Recurrence Plots: In our fluid flow measurements, we are particularly interested in searching for occurrences where the flow field repeats a similar pattern that the flow exhibited earlier; these recurrences are evidence for the possible existence of unstable periodic orbits in the flow. To search for recurrences in a quantitative way, we compute a "distance" (more precisely, an energy norm) between the velocity fields at time t and at time t+τ, which, in discrete form suited for our computed velocity fields is given by:

$$\sum \Delta\vec{v} \bullet \Delta\vec{v}$$

where: $\Delta\vec{v}(t,\tau) \equiv \vec{v}(t) - \vec{v}(t+\tau)$

This energy norm can be represented on a single time delay plot (or recurrence plot) where a color scale can be used to represent the size of the energy norm. Regions of the recurrence plot where the energy norm is small are of particular interest since this suggests that the two flow fields are nearby; the corresponding time difference τ *may* be related to an unstable periodic orbit in the flow dynamics. Convincing evidence of the presence of such unstable periodic orbits in an experiment on turbulent flow would represent an important advance in our understanding of turbulence.

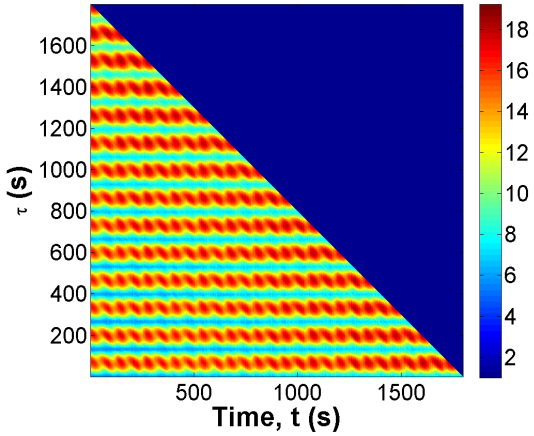

Figure 3: A recurrence plot comparing the flow at time t to the flow at time t+τ. Red regions correspond to dissimilar states and blue/green regions correspond to similar states. One can clearly see that this data is oscillatory, corresponding to a *stable* periodic orbit solution.

Further reading:

OSIV provides a large number of ways to tune and to optimize the analysis; for full details and downloads, see http://osiv.sourceforge.net/

Acknowledgements:

We would like to thank Daniel Borrero (Georgia Tech), who helped in organizing the 2010 Hands-On School in Cameroon, for writing the first version of this document.

We would also like to thank Jon Paprocki (Georgia Tech), who wrote the initial version of the Matlab scripts.