The Ion Doppler Spectrometer: Its Hardware, Software, and Usage

Rian Chandra

University of Washington

Abstract

Basic Operations Super TLDR:

1. Assume: The IDS fiber or fibers are inserted, locked, and oriented as required, and the the spectrometer is dialed to the wavelength of interest (usually 464.97nm).
2. Check to make sure that the camera is turned on (lights on back are on), and that the camera computer can communicate with it (open either Phantom Camera Control program on the IDS camera operator computer, although the older one (shortcut on desktop) is somewhat more intuitive). Check to make sure that the Framerate, Resolution, and Exposure are correct, in that order.
3. Open Matlab. Run

   `T:\RChandra\Phantom\PhantomStalker`

4. Run PhantomStalker again. In theory, it should automatically pull the shot number from the tree, convert the cine from the camera, and save it, and the converted file. Sometimes you need to restart it if it has a hard time. It won't work if a baseline shot doesn't exist, probably.
5. Exposure and other parameters can be set in Phantom Camera Control as needed.

1 Setup and Calibration

1.1 Hardware Overview

The IDS spectrometer is an f/8.5 Cerny-Turner configuration grating spectrometer on loan from the University of Hyogo. The two $\approx$ 2m long linear bundles of 36 fibers each terminate in an Edmond Optics "Micro Video" fish-eye lens. Each channel is separated by 2.95°. The output of the spectrometer is optically coupled to a Phantom V710 fast camera.

1.2 Calibration

1.2.1 PIX_SP

Why: because we need to convert size and shift on the CCD to wavelengths.

How: Setup one fiber on the test stand as pictured. Setup motor box (plug in connections). Set motor speed to XXX nm/s. Dial spectrometer to 435.6nm. Set the camera to continuous recording, trigger it, and immediately turn on the motor, running down (–¿ lambda decreases as time goes forward). You should see the main line, and the a few seconds later a secondary, weaker line (and potentially a third line). Stop the movie after the second line has past. XXXX requires knowledge of Ph692.

1.2.2 PEAKS

Why: Because we need to know where the centers and zero temperature widths are, relatively, between all the fibers. Particularly important if using both fibers in the midplane.

How: Set a fiber in the correct orientation in the test stand. Dial spectrometer to XXX 435.6nm XXX, set Ph692 to continuous recording, trigger, and record for several seconds. Save the file. Do this for each fiber.

1.2.3 Backlight

Why: To confirm that the fiber lens is correctly focused.

How: Use the arts and crafts box, which lives above the cabinet which holds old PMTS and is next to the optics cabinet. The string is approximately the length of the major radius. Set up the fiber in the holder. Place the box directly in front of the fiber, one string length away. Remove the spectrometer top and middle panels from the entrance slit. Place the 1ft long, 2in diameter black laser (sometimes used for Thompson calibration) inside the middle panel, raised by 1-2 in. Adjust as necessary so that circles of light appear inside the box, adjust the lens on the fiber to focus these circles. If necessary, rapidly swinging the laser up and down can be used to align the fiber holder in an absolute vertical position. Note: Critical care should be taken whenever the spectrometer panels are removed, and when working inside it.

1.2.4 Angles

Why: This information goes into calc_geometry to find the impact parameter of each chord.

How: Setup the fiber holder on the stand, opposed to the mercury lamp, and zero the angle ticks. Find the maximum and minimum angle that illuminates the fibers. Enter this range into calc_geometry_5.

1.3 Setup

Directly after calibration is finished, the fiber holders should be carefully returned to the re-entrant ports, maintaining vertical alignment. Markings on the sides of the holders, and the port entrances should indicate proper alignment. The holders should be inserted with a small amount of space at the very end (ie: not pressed directly up against the glass).
Port options and names:

- Upper Mohawk 6: Used for most of HIT-SI data, views upper half of the midplane.
- Lower Mohawk 6: Second component of "Dual Multi-Chord IDS" configuration, views coore-sponding impact parameters below geometric axis as Upper Mohawk 6.
- 71 degree radial port: Central fiber in array looks directly through geometric axis.
- Coherant Fiber Ports: Used to take movies with coherant fiber bundle.
- Poloidal Ports: Oppositly facing poloidal ports, used on HIT-SI to absolutely specify poloidal rotation.

1.4 Calibration Codes

The calibrationMAIN code is relatively well documented. It takes in the calibration movies and some additional information and stores the reference values the tree for a specified region of shots, or the model tree shot (-1). The values collectively saved by calibrationMAIN are organized

by fiber channel. PEAKS column one links channel number to row index. What Calibration Saves to the Tree

- **CAL_LAMBDA:** this should be the wavelength, during data acquisition, that the calibrated line will be set to. When doing this, the assumption that the spectrometer grating behaves roughly linearly is made. (IE: that PIX_SP is the same, even at a slightly different wavelength). This is used to calculate the expected instrument temperature for the lines being viewed.
- **LAMBDA:** The wavelengths, in meters, of the lines being viewed for data acquisition. This is necessary for predicting the relative locations of the observed lines on the CCD. It is also necessary when calculating temperature.
- **VOLTAGE**The voltage of the Image Intensifier. Not technically depreciated, but rarely used.
- **MASS:** Ion mass of the viewed lines, in AMU. Necessary for calculating velocity and temperature.
- **PEAKS:** In the following column order: Channel number, the x center (necessary for obvious reasons) and y center (delta y gives us velocity)., the x width of the gaussian (which is currently only used to improve the initial guess for data fitting), and the y width (the change in which gives us temperature).
- **IMPACTS:** Generated by calc_geometry5, impacts gives the impact parameter corresponding to each channel. This is specific to whichever port and orientation the fibers are in, and so may change between calibrations.

2 The Tree

2.1 How the Tree Works

Code References: PhantomStalker, SaveToTree3, LoadParams, BatchCorrect2, The MDSplus online tutorial.

The HITSI3 tree (and the ANALYSIS3 subtree) store the experimental and calibration data for IDS, and now the raw data from the movies as well. Data is stored in formats specific to MDSplus. The tree structure can be visually navigated via the Traverser program, installed on most lab computers. This is a useful way to get a feel for the layout. The JScope program is used to display signals from the tree. This is not terribly useful for IDS at present, although eventually the raw movies will be viewable through this.

Each shot has its unique hitsi3, analysis3, and imaging3 tree (the latter two can be accessed from the hitsi3 tree). Data is stored in Nodes, and is usually Numeric type. Nodes can either be referenced from the top of the tree, or through a tag, such as /IDS_MASS. Any data entry, or node traversing will be referenced from the current node.

2.2 Accessing Data in the Tree

There are three different methods of accessing data from the tree, all of which are slightly nuanced. The connection object (Thin Client) is the newest way to connect to the tree. It has been found to be the most straightforward way to do basic data entry and access. Modification of the tree structure itself is best accomplished using the Tree Object (Object Oriented MDSplus). Performing mathematical operations, such as sihi_smooth, or more complex TDI expressions, can be accomplished in the Thin Client, but is easiest in the mdsconnect (TDI expression) framework.

- **Thin Client:** loadParams, saveToTree3, both in

  `\NewCodes`

- **Connection Object:** PhantomStalker in /RChandra, saveToTree3 in
  IDS
  Calibration.

- **mdsconnect:** Not currently used in any up-to-date IDS code,

2.3 Further Tree Notes

Note the difference between opening a tree, and opening it for EDIT. This should only be done when absolutely necessary.

Environment variables and path: Check a computer which can successfully access the tree, and make sure that the environment variables on the machine you are using include analysis3_path, hitsi3_path, and imaging3_path. Remove the 3 if looking at hitsi data. Matlab also needs to know where the MDSplus libraries are stored. Note that import in matlab will unfortunately not throw an error if the imported library is not found. The MDSplus tutorial has the correct path locations. This may require modifying the javaclasspath.txt file, or the librarypath.txt. It may be easier to manually add the paths in the matlab GUI, and then use savepath to permanently add the files to your path on startup.

Directly on import (either from the Tree object or the Connection object), the data will be of type MDS_(float32/uint/float32array/etc). If all of the MDSplus matlab files are on the path, NATIVEvalue from MDSplus or double from matlab will convert to a usual type.

When adding data into the tree, clear the node first (see SaveToTree3)

When adding arrays, you have to add the data as MDSargs(data), or you will be instead adding squeeze(data), and everything will be a vector.
See PhantomStalker and SaveToTree3 for examples of how to move around in the tree.

3 Data Collection Software

3.1 The PCC

Assuming the camera is connected via the fiberoptic ethernet adaptor, it can be controlled from the IDS camera operator computer by way of the Phantom Camera Control software, shortcutted from the desktop. -¿ Acquisition -¿ Setup and Recording opens a screen where the current camera image can be viewed in real time, and acquisition parameters can be set. To prepare the camera for data acquisition, click Continuous Recording, and the camera will wait for the server to send out the trigger signal.
Important Notes on the PCC:

- The framerate should be changed to be some integer multiple of the injector frequency. The exposure should be maximized given this frame rate. Do not attempt to optimize based first on exposure. This will lead to beat frequencies between the frame rate and the injector period.

- After it has been triggered, closing the acquisition window will open a viewer for the currently recorded cine. It is important that if any image processing parameters are changed, they are reset to the default before the cine is converted into a .mat file. While a fix has been implemented into PhantomStalker, these changes can theoretically be propagated through, and the .mat file will not be useable. Explanations of how the image processing parameters affect the movie can be found in the Cine Documentation.

- If there are too many PostTrigger frames, data processing will be very slow.
- Occasionally, the cine will appear to have a completely black background (intensity zero). There is no fix for this if it happens during data acquision. Occationally during calibration, however, different computers can interpret the same cine slightly differently. This can be corrected by confirming that the bit depth is an acceptable range for both the sending and receiving computer. The lab computer seems less prone to this, but the IDS camera operator computer can sometimes see the same cine, and have this issue.

3.2 PhanomStalker

The PhantomStalker Matlab script replaces an older labview program, whose job was to grab and convert the .cine files on the camera, and save them as .mat files on the movie hard drive. The new script trims the movies to the shot length, and saves them in the tree. It is compatible with PDC and regular shots.

To run it, all of the files in /PhMatlabSDK must be on the path, except /PhMatlabSDK/bin/win32. That folder must be explicitly excluded.

PhantomStalker requires several precompiled .dlls, which in turn require a compatible C compiler. It has proved extraordinarily difficult to find and install such a compiler, which must be compatible with the OS and Matlab version. For this reason, if the IDS camera control computer must ever be relocated, it is suggested that the physical computer itself be moved along with it.

While the PhantomStalker program can run essentially autonomously once started, it does frequently have difficulty finding the correct shot number. A patch has been added for this, but sometimes the program will need to be restarted a few times to work. This is particularly noticeable PDC, where the period between shots may be shorter than the time necessary to process the cine data. A fix has been implemented to prevent conversion of frames far after the injector shutoff.

3.3 Cine File Structure

The Cine file structure is the proprietary file format of the Phantom camera movies. Significant documentation exists for both the the file format itself (Cine File Format) and the matlab commands to interface with it (Phantom SDK) both in /PhMatlabSDK/doc.

On the lowest level, each .cine file is broken up into the header, which holds all of the information about the state of the camera settings at the time of collection, and the image processing settings that have been applied post hoc, followed by the frames themselves. Inside each frame, each pixel intensity is stored as a XXXXXX12XXXXXX bit value, broken up across two bites. Note: there occasionally appears to be a scalar offset on the intensity values, when compared to the PCC program.

On the higher level, the most important concept to keep in mind is the difference between frame time, trigger time, exposure and interval. Frame time is the timestamp associated with the beginning of the frame. Trigger time is the time that has elapsed between the start of the frame in question, and the time that the injectors turned on (or, the time that the server sent out the PHANTOM TRIGGER event). Exposure is how long the shutter was open for, which is the same for all frames in a given cine. Interval is the length of the frame. Importantly, Interval is slightly longer than exposure because some time is necessary each frame to reset the CCD frame rate sets interval, not exposure. For the purpose of comparing to simulation, and producing phase portraits, IDS shifts the frame time so that it is zeroed to the start of the shot, and so that the timestamp is in the middle of the frame. Once turned on, the camera is recording frames continuously, once the trigger is sent out from the server, the camera, which will be in the middle of recording a frame, starts the cine file at the next frame, labels it with frame time zero, and records how long before that time the trigger happened. IDS corrects this by adding the trigger time to the frame time,

and 1/2 of the interval time.

Access of data in the cine can happen several ways. Previously, the cine was brought in from the camera with LabView, and turned into a .mat file with a python script from NSTX, stored on the server. Currently, we make use of the proprietary Matlab SDK. As noted previously, this relies on several pre-compiled DLLs, the used of which is tricky at best. The SDK is well documented, although care should be taken as newer versions of the SDK seem to have slightly different syntax (they are somewhat more object oriented). Note that some functions do not have wrappers built in. See GetCineAuxData for an example of this. There are several quirks to data access using the SDK, particularly surrounding the variable types that it uses, and converting them back into primitive types. All of these are well documented, but it can be hard to find. PhantomStalker should have relatively clear examples of how to get a handle to a cine from the camera, and then pass that handle around to various methods to extract data.

4 Data Processing

4.1 Batch_Correct2

Batch Correct is the primary data processing tool used by IDS. Each .mat file is loaded in from /PhantomMovies (although in theory it could be loaded in from the server: this should be tested), and filtered using bd filter. The processing of the raw data requires it to pull down calibration data from the server (loadParams), along with several user-supplied values and modifications. It fits gaussians to each fiber, in each frame, of each movie, and uses the y-width and center to calculate temperature and velocity. There are many places where manual modifications have been added in to improve fitting accuracy, etc, such as the window size around each gaussian, and values having to do with modifications on the calibration data. It also loads in additional information from the server about the injectors.

4.2 BD Filtering

See Brian Victor paper for the mathematics behind the use of SVD decomposition (Biorthogonal-Decomposition), and Aaron Hossacks Ph.D thesis for IDS applications. For our purposes, it is sufficient to consider it solely a filtering tool. Note that Dynamic Mode Decomposition may be a future area of filtering exploration.

4.3 Lsqcurvefit

Using the calibration values in PEAKS, a window is fit around every fiber, centered on x0,y0, with some window xWing and yWing, set in loadParams, for every frame. In gaussFit2D, extra methods are applied to get the best initial guess possible, such as shifting the window around slightly, and modifying x0,y0 to correct for bumping the spectrometer. Note that these changes are not what the final fits will compare themselves to in calcPhysics. The lsqcurvefit package performs the nonlinear least-squares minimization of the data onto a trial function (the gaussian), given the initial guess. A slightly modified version checks the error on this fit, an option selected in loadParams as well.

5 Data Analysis

5.1 Compare_Plots_Multiline

This plotting routine, frequently used first as a sanity check on the data, produces the color square surface plots seen on some posters in the hall. It can display the analyzed quantities stored in the .dat file, and do some analysis on its own. Which fiber array is being plotted can be changed, as can whether the data is displayed by impact parameter or channel number, and time point or time point number. While the surface plot style may not be as useful to the uninitiated, with some practice it is a quick and easy way to check if the data overall looks reasonable.

5.2 Multiplot