Solving for the pointing model of CBASS

1. Introduction

As we know, the pointing model of a single-dish telescope is absolutely crucial for mapmaking purposes. Given its importance and the relative (in)frequency which we run pointing schedules, the analysis process is not as automated as our regular pipeline – user input is crucial to make sure we get the most accurate pointing models. What follows is a tutorial on how to collect and analyze the CBASS pointing data – both optical and radio pointing data. (For more detail on the model parameters themselves, see my other writeup on the wiki http://astrowiki.physics.ox.ac.uk/CBass/CommissioningDocuments/opt_point.pdf)

2. Optical Pointing

2.1. Gross Pointing model

2.1.1. Getting things ready for the schedule

Determining the pointing model from scratch can be quite cumbersome given the small field of the view of the optical telescope, but it is amazing the amount of accuracy one can get for a few parameters just by tracking a single source as it rises, transists, and sets. The first thing you must do is select a suitable star, one that is bright and is rising at the time you wish to run your schedule. You can select such a star in cbassViewer in the "Utilities" tab by opening the Star Plot. In the Star Plot, load the catalog in gcpCbass/control/ephem/starList.cat, and this will put up the stars that are currently up. Change the LST field to correspond with when you wish to run your track, and select a bright star that's in the east and at an elevation of less than about 40 degrees. For this example, let's use AlpAql.

Next you will need to find the az and el offset that will put this star within the field of the view of the telescope. To do this, you will need to connect to the Frame Grabber on the control computer (from cbassViewer go to Utilitites and select Frame Grabber). In the Frame Grabber Control window, select "Channel 1", and in the Pointing Control window, click on "Next Frame" to see the image through the optical telescope. If there is nothing in the field, go to Frame Grabber Control and "Take Flatfield" to normalize all the pixel gains. Next you should track the desired source, and once the telescope thinks it's tracking, click "Next Frame" to see if it's in the field of view. *Make sure the telescope is using the optical pointing parameters and not the radio pointing parameters, by typing "model optical,*

ptel=1" in cbassViewer. If it is, great, you might not have to do this step (see §2.4)! If not, you'll have to play around with the az/el offsets until it is. Once it's in the field of view, in the Pointing Controls window click on "Find Peak", and if it has found your star, click on "Center". Now you **must** record the az/el offset of your star (Look in Status Page window for az/el Offset column). Record these values. While you're at it, you should find out how bright your star is in SNR, so type "print $imstat(snr)" in the cbassViewer window and record this value as well.

Next you will have to run a schedule to track only this source: so you have to run gcpCbass/control/sch/pointing/point_cbass_single_source.sch. The arguments for this track are (source, lststart, lststop, npoints, azOffset, elOffset, snr) where snr is something smaller than the value you recorded on the star. Choose npoints to be in the 10-15 range, but it doesn't matter for this schedule. This schedule will go to the desired offset on the source, and just track it as it goes across the sky, centering ever few seconds.

2.1.2. Analyzing your data

To analyze your data, you must first make sure you're data is good. Fits files are written out for each image captured by the frame grabber, so the best way to check is to look at all your fits files to make sure they have a star near the center of them. If my track ran from an UTC of 18 to 23 on Feb 30, 2011, to check the images I call the matlab function "checkImages('20110230_18*')" to make sure there's a star in the images for the 18 hour block. This will make a series of plots of the fits images, with the time as the title of the plot. If there was a time from 18:35 to 18:40 where the star was not in my field, I make a note of it. I repeat this for the different hours that my track ran, and note down the times when the data is not good. To read in the data, I use read_arc on the time spans that were good, concatenate the data together, and then I do a "dc = framecut(d, d.array.frame.features>0);" to get only the data where the star was centered.

Next I run optical_pointing(dc, '20110230'), and this will automatically fit the pointing model and tell you what they should be on exit. I copy the four lines that are printed at the end into the gcpCbass/control/conf/pointing.init file, and copy those into the control system and wait for the next night.

2.2. Refined Pointing model

The solution you get from tracking one star is usually not global because it only covered a certain part of the sky. This means that if you view a star on one side of the sky and then go to another part of the sky, chances are you won't find them both to be in your field of view. However, if you have a star centered in your field of view, and go to a nearby star, chances are that one will be centered. You can do this from star to star and in this way make your way across the sky and always have them in your field of view (as long as you don't zero your offsets between stars). The second step in refining the pointing model uses this method, and uses schedules called auto_point_cbass_lstXX.sch, where XX corresponds to the closest LST to the start of your track. For these observations, determine when you're going to run the track, open the corresponding schedule, and see the name of the first star on the list (i.e., PhiPer for _lst07.sch). Track that source, and make sure it's in your FOV – Center it if needed, and record the az/el offset. Next run your schedule, setting the first argument to 15, and the second/third arguments to the az/el offsets you just recorded.

To analyze the data, repeat the process from the previous section.

2.3. Global pointing model

You are now ready to do a full pointing model. To do this, run gcpCbass/control/sch/pointing/auto_point_cbass_3min.sch(15, lststart, lststop). To analyze the data, repeat the process in §2.1.

2.4. Which step am I in?

To determine which of the three stages above your telescope is in, you should get the Frame Grabber setup, open up a starPlot, and see which stars are up. Make sure you have the optical model loaded ("model optical, ptel=1"), and issue a track source command. If the source is not in the FOV of the telescope, you're at the very first step and need to get a "Gross pointing model". If it is, pick a few other sources throughout the sky and go to each one of them. If they're all in your FOV, you're at the very last step. If not, you're likely in the middle one.

2.5. I had to remove/install the optical camera

If you had to remove the optical camera and re-install it, you have to characterize the orientation angle of the camera. You'll do this by first getting an object in the field of view of the camera. Then offset the telescope by a few arcminutes in azimuth, and see where the source went. Using your geometry skills, you can then figure out the orientation angle of the telescope. To test whether the determined orientation is correct, input it into the control system via cbassViewer as "setOpticalCameraRotation angle=XXX" – you'll see the axis in the frame grabber image rotate. With the star in the field of view, hit findpeak followed by center. In your next image, the star should nominally be centered – if it is not, the orientation angle you put in is not correct. I suggest checking your geometry and adding offsets of 180 degrees. Repeat the procedure until you get the proper camera orientation. Once you have this, you **must** modify the corresponding line in the pointing schedules to reflect the new orientation. The other parameters (camera FOV, etc) should not change when you put the telescope on/off.

3. Radio Pointing

The optical pointing model and the radio pointing model share 5 of the 9 terms (encoder zeros and tilts), so for the radio pointing we're only fitting for the flexure and collimation terms of the radio axis. Select an LST range over which to take your data, and queue in gcpCbass/control/sch/radio_pointing_bright.sch(lststart, lststop). This schedule does pointing crosses of 10degress in azimuth and elevation over bright sources (cyga, casa, taua). Once the data is taken, read it into Matlab (using read_arc), then run apparentAzEl on your data to obtain the apparent locations. Next you should run radioPointing(d, 'scan', 'NAME'), where NAME is whatever you want it to be.

This program will fit gaussians to the peak of each scan, and plot up the total intensity channel data. After each set that is plotted, it will prompt you whether the fit was good. If it is a good fit (there is high signal-to-noise and the fit gaussian goes through the peak of the data for **both** az and el plots), you hit 'Y', and enter. It will go through this for all the scans in your data set, and once it's done the program will fit those four parameters of the pointing model to the data. Once it is done, it will again print out the four lines that should be copied to the pointing.init file. To load the file immediately, just "schedule" it from the cbassViewer. You should always check to see if the solutions are sensible: for example, pointing terms on the order of tens of degrees are likely bogus.

Once complete, I would load in the new pointnig model, and take another hour or two

of radio pointing data to make sure the model is a good fit. But otherwise, that's all there is to it!