Table of Contents

RICH mirror alignment monitoring via software (Online and Offline) by the use of real data *Legacy Information*......1

Offline mirror alignment Instructions......1

Most Recent Instructions......1

Previous Instructions......3

How to submit the Offline Mirror Alignment via LSF......5

Ancient Instructions......7

Working directory......7

Data (This example was written for Collision 11 Mag Up)......7

Running......8

Scenarios......8

Pathway to RICH Mirror Alignment......9

Q & A......9

Old Information, Postponed Indefinitely, or Obsolete......10

Current list of possible explanations for and differences in RICH1......10

Problems to be solved for past (i.e. 2012) RICH mirror alignments......11

2015: Question about necessity of separate Mag Up and Mag Down constants......11

Rich Alignment person on shift at CERN (to be contacted 8-21h only)......14

2016......14

2015 Offline EM Magnet Down data......14

Implementation of Code in the Online Framework (Claire should give information for Rich/RichMirrorAlignmentOnline v0r3 and beyond in a separate section. Please preserve this section)......14

Online Alignment Framework Sequence Information (based on Rich/RichMirrorAlignmentOnline v0r2. Claire should describe elsewhere the situation for v0r3 and beyond. Please preserve this for now)......14

How to run the Online Mirror Alignment Tests for Rich/RichMirrorAlignmentOnline v0r2 (Claire should write this elsewhere for v0r3 and beyond. Please preserve this for now)......16

Coombes Automated Alignment Procedure......18

RICH mirror alignment monitoring via software (Online and Offline) by the use of real data *Legacy Information*

For current information, see LHCbRichMirrorAlign

Offline mirror alignment Instructions

Most Recent Instructions

We no longer do Offline alignments, but there may be information here that could be helpful to you if you can't find it anywhere else.

▶ Show... ▼ Hide
Instructions so that anyone can do an Offline alignment:

- - First, you should have an lxplus account and an installed Grid proxy. See: https://twiki.cern.ch/twiki/bin/view/LHCb/FAQ/Certificate for more info on the latter.

- - Second, you need to edit your `~/.gangarc` file. There are two (commented) lines in this file that start with `"#NumWorkerThreads"`

- ---- After **both** of the commented lines [I am not sure how Ganga picks up the right one] please add this line (uncommented):

```
NumWorkerThreads = 9 # you could even go higher (I use 12), but at some point Ganga will complain
```

- ---- Consider changing/moving your default "gangadir" from your home area to your work area (hopefully you have a work area, if not ask about how you can get one). For example:

```
#gangadir = /afs/cern.ch/user/p/pnaik/gangadir
gangadir = /afs/cern.ch/work/p/pnaik/gangadir
```

- - Instructions below are provided for `csh` type shells. (For `sh` type shells, change the alias command appropriately)

- - **NOTE: You may be working with the nightly builds. In the early hours of the morning CERN time the builds break because they are being rebuilt. Always specify the day of a nightly build if you plan to run the code, e.g. if your code will run overnight CERN time into Wednesday, then instead of "-nightly lhcb-head" use "-nightly lhcb-head Tue".**

- ---- Open four clean shells using `ssh -Y lxplus.cern.ch`. The following is assuming you want to work with the HEAD version. If not, change "HEAD --nightly lhcb-head" to a stable version, like "v5r5" for example, and edit your settings file (explained later) accordingly.

- ------- One of these is to be the "development (DEV) shell." You only need this one if you plan to edit the code. In a `csh` shell:

```
alias PanoptesDev 'SetupProject Panoptes HEAD --build-env --nightly lhcb-head'; PanoptesDev
```

- ---------- This will move you into **your** `Panoptes_HEAD` cmtuser directory. From here you can getpack any code if you need to modify any of it. Otherwise it will use the precompiled code from the nightly build.

- ------------- **NOTE, Jeremy discovered that the precompiled code is not being picked up from the nightlies. We expect precompiled code to be in $RICHMIRRCOMBINFIT/$CMTCONFIG/ and $RICHMIRRALIGN/$CMTCONFIG, but it was not (or maybe this is only true for production versions, not HEAD). Anyway, this has to be fixed, but for now please getpack the main code packages locally. From inside your `Panoptes_HEAD` cmtuser directory, run these two lines:**

```
getpack Rich/RichMirrCombinFit; cd Rich/RichMirrCombinFit/cmt; cmt br cmt config; cmt br cmt make
getpack Rich/RichMirrAlign; cd Rich/RichMirrAlign/cmt; cmt br cmt config; cmt br cmt make; cd -
```

- ------------- Then **if need be** you can edit the code and recompile as much as you need to locally, keep it **updated** locally, and commit changes (FOLLOW THE LHCB SOFTWARE RULES PLEASE).

- ------- One of these is to be the "runtime (RUN) shell." In a `csh` shell:

```
alias PanoptesRun 'SetupProject Panoptes HEAD --nightly lhcb-head'; PanoptesRun
```

- ---------- create a **working directory** (if it does not exist yet) and `cd` to it, e.g. for 2015 Rich2 Magnet Down, but of course use and choose your own directory:

```
mkdir -p /afs/cern.ch/user/p/pnaik/RichAlignment/2015/2015Down_R2; cd /afs/cern.ch/user/p/pnaik/R
```

- ---------- you will also a need a settings file, e.g. for 2015 Rich 2 MagDown:

```
cp /afs/cern.ch/user/p/pnaik/RichAlignment/2015/2015_R2_EM_Down_1250k_ev/RichMirrorAlignmentGanga
```

- ----------- e.g. for 2015 Rich 1 MagDown:

```
cp /afs/cern.ch/user/p/pnaik/RichAlignment/2015/2015_R1_EM_Down_1250k_ev/RichMirrorAlignmentGanga
```

You should not need the runtime shell, but good to know how to do this if you ever need to run/test pieces of code manually

- ------- One of these is to be the "Ganga shell." Please ideally open this only on a computer that you do not have to disconnect from the internet. (you could use a screen session, though we don't know how you would save the output, maybe Claire knows)

```
alias GangaRun 'SetupProject Ganga v600r43 pytools'; GangaRun
```

- ---------- `cd` to your **working directory** and create a 7-day renewable AFS token, e.g. for 2015 Magnet Down:

```
cd /afs/cern.ch/user/p/pnaik/RichAlignment/2015/2015Down_R2
kinit -r 7d
```

- ---- Note that if you plan to do any work or run any scripts which will operate after 23:59 CERN time, then the nightlies will be broken until they are rebuilt. To prevent broken code, make sure to specify which day's nightly build you would like to use before you run code or a script, i.e. if you want to make sure you use Tuesdays's nightly build, change "lhcb-head" to "lhcb-head Tue".

- ---- Now edit your RichMirrorAlignmentGanga _Collision15_settings_R2.py file. The five lines you need to change (to what you think is appropriate) look like:

```
home  =  "/afs/cern.ch/user/p/pnaik" # Point this to *your home directory*

setupProjectPanoptesOptions = "--nightly lhcb-head Sat" # Safest to specify a day that you know i

workDir = "/afs/cern.ch/user/p/pnaik/RichAlignment/2015/2015Down_R2" # Point this to *the full pa
```

```
thisCase = "2015Down_R2" # A label for this particular alignment

<!--
AddedDB = ["Alignment2015.db", "MDCS-RICH1-SecondScans-26052015.db", "ResetHPDAlign-13062015.db",
-->
```

- ---- Before you run Ganga, if you wish to log all of the lines that show up on the screen in your Ganga session do the following: (**PLEASE NOTE, AFTER you exit Ganga, end the logging by typing exit or CTRL-D; otherwise the script command will continue logging**)

```
script YYYYMMDD-HHMM.log
```

- ---- Also, make a long grid proxy (and don't renew it until you have to) [**if and only if** you have special rich calibration privileges (Paras, Claire), use `lhcb-proxy-init -g lhcb_calibration -v 120:00` instead]

```
lhcb-proxy-init -v 96:00
```

- ---- Run `ganga`. In the Ganga shell, start the engine:

```
RichInfoName = ( './RichMirrorAlignmentGanga_Collision15_settings_R2.py' ) # e.g. for 2015 Mag Do
execfile( '/afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/job/Rich
```

- ---- Ganga should then perform the alignment. Note that the file RichMirrorAlignment.py is the Offline alignment driver script, which reads in the _**settings**_.py file and performs the alignment. Paras tries very hard to keep his local version of the script in sync with the one in the HEAD version of `Rich/RichMirrorAlignmentGanga`.

- ---- **NOTE: Whether the alignment finishes properly or it crashes, do not continue in the same Ganga session! Exit ganga using exit() or CTRL-D. Then start `ganga` again.**

Always remember to end the script command if you have run it and exited Ganga, or it will continue to store everything that shows up on the screen in your file until you exit your shell.

Previous Instructions

▸ Show... ▾ Hide

First, you should have an lxplus account and an installed Grid proxy. See: https://twiki.cern.ch/twiki/bin/view/LHCb/FAQ/Certificate for more info on the latter.

Second, you need to edit your `~/.gangarc` file. There are two (commented) lines in this file that start with `"#NumWorkerThreads"`

After **both** of the commented lines [I am not sure how Ganga picks up the right one] please add this line (uncommented):

```
NumWorkerThreads = 9 # you could even go higher (I use 12), but at some point Ganga will complain
```

--- Instructions below are provided for `csh` type shells. (For `sh` type shells, change the alias command appropriately)

NOTE: You may be working with the nightly builds. In the early hours of the morning CERN time the builds break because they are being rebuilt. Always specify the day of a nightly build if you plan to run the code, e.g. if your code will run overnight CERN time into Wednesday, then instead of "-nightly lhcb-head" use "-nightly lhcb-head Tue".

Open three clean shells using `ssh -Y lxplus.cern.ch`. The following is assuming you want to work with the HEAD version. If not, change "HEAD --nightly lhcb-head" to a stable version, like "v5r5" for example.

One of these is to be the "development (DEV) shell." You only need this one if you plan to edit the code. In a `csh` shell:

```
alias PanoptesDev 'SetupProject Panoptes HEAD --build-env --nightly lhcb-head'; PanoptesDev
```

------------ This will move you into **your** `Panoptes_HEAD` cmtuser directory. From here you can getpack any code if you need to modify any of it. Otherwise it will use the precompiled code from the nightly build.

NOTE, Jeremy discovered that the precompiled code is not being picked up from the nightlies. We expect precompiled code to be in $RICHMIRRCOMBINFIT/$CMTCONFIG/ and $RICHMIRRALIGN/$CMTCONFIG, but it was not (or maybe this is only true for production versions, not HEAD). Anyway, this has to be fixed, but for now please getpack the main code packages locally. From inside your `Panoptes_HEAD` cmtuser directory, run these two lines:

```
getpack Rich/RichMirrCombinFit; cd Rich/RichMirrCombinFit/cmt; cmt br cmt config; cmt br cmt make
getpack Rich/RichMirrAlign; cd Rich/RichMirrAlign/cmt; cmt br cmt config; cmt br cmt make; cd -
```

--------------- Then **if need be** you can edit the code and recompile as much as you need to locally, keep it **updated** locally, and commit changes (FOLLOW THE LHCB SOFTWARE RULES PLEASE).

One of these is to be the "runtime (RUN) shell." In a `csh` shell:

```
alias PanoptesRun 'SetupProject Panoptes HEAD --nightly lhcb-head'; PanoptesRun
```

------------ create a **working directory** (if it does not exist yet) and `cd` to it, e.g. for 2010 Magnet Down:

```
mkdir -p ~/RichAlignment/2010/R2MD; cd ~/RichAlignment/2010/R2MD
```

------------ you will also a settings file, e.g. for 2010 Magnet Down:

```
cp /afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/job/settings/Ric
```

------------ Please note that this file is in **Paras's** cmtuser area. He tries to keep this in sync with the HEAD version of `Rich/RichMirrorAlignmentGanga`.

One of these is to be the "Ganga shell." Please open this only on a computer that you do not have to disconnect from the internet.

```
alias GangaRun 'SetupProject Ganga v600r43 pytools'; GangaRun
```

------------ `cd` to your **working directory** and create a 7-day renewable AFS token, e.g. for 2010 Magnet Down:

```
cd ~/RichAlignment/2010/R2MD; kinit -r 7d
```

------ Note that if you plan to do any work or run any scripts which will operate after 23:59 CERN time, then the nightlies will be broken until they are rebuilt. To prevent broken code, make sure to specify which day's nightly build you would like to use before you run code or a script, i.e. if you want to make sure you use Tuesdays's nightly build, change "lhcb-head" to "lhcb-head Tue".

Now edit your RichMirrorAlignmentGanga _Collision10_settings_R2MD.py file. The four lines you need to change (to what you think is appropriate) are:

```
home  =  "/afs/cern.ch/user/p/pnaik" # Point this to *your home directory*

setupProjectPanoptesOptions = "--nightly lhcb-head Sat" # Safest to specify a day that you know i

workDir = "/afs/cern.ch/user/p/pnaik/RichAlignment/2010/ParasR2MD" # Point this to *the full path

thisCase = "R2MD" # A label for this particular alignment
```

------ Before you run Ganga, if you wish to log all of the lines that show up on the screen in your Ganga session do the following: (**PLEASE NOTE, AFTER you exit Ganga, end the logging by typing exit or CTRL-D; otherwise the script command will continue logging**)

```
script YYYYMMDDHHMM.log
```

------ Also, make a 96 hour grid proxy (just to be safe)

```
lhcb-proxy-init -v 96:00
```

------ Run `ganga`. In the Ganga shell, start the engine:

```
gridProxy.renew() # Always do this right after you start Ganga!
RichInfoName = ( './RichMirrorAlignmentGanga_Collision10_settings_R2MD.py' ) # e.g. for 2010 Mag
execfile( '/afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/job/Rich
```

------ Ganga should then perform the alignment. Note that the file RichMirrorAlignmentGanga.py is the Offline alignment driver script, which reads in the **_settings_**.py file and performs the alignment. Paras tries to keep this script in sync with the one in the HEAD version of `Rich/RichMirrorAlignmentGanga`.

NOTE: Whether the alignment finishes properly or it crashes, do not continue in the same Ganga session! Exit ganga using exit() or CTRL-D. Then start `ganga` again.

Always remember to end the script command if you have run it and exited Ganga, or it will continue to store everything that shows up on the screen in your file until you exit your shell.

How to submit the Offline Mirror Alignment via LSF

▸ Show... ▾ Hide
Example (BEING TESTED NOW):

Set up everything as before in a new working directory. Change the settings file to refer to this new working directory.

The following instructions are for `tcsh`, please convert accordingly if you use another shell (sorry):

Add the following to your .login file:

```
setenv X509_USER_PROXY /afs/cern.ch/user/p/pnaik/private/.grid.proxy
```

where `p/pnaik` should be changed to your username intitial and username, of course.

This forces your grid proxy to be stored in your AFS area instead of on /tmp. This is necessary for the LSF machines to see your grid proxy. Why this is not default, we don't know.

After this you will need a new shell, or just continue by setting manually setenv X509_USER_PROXY $HOME/private/grid.proxy

Then `cd` into your working directory

```
mkdir Ganga_LSF

cd Ganga_LSF
```

Then make the following `submit.csh` shell script:

```
#!/bin/tcsh
setenv X509_USER_PROXY /afs/cern.ch/user/p/pnaik/private/.grid.proxy
setenv MIRRWORKDIR /afs/cern.ch/user/p/pnaik/RichAlignment/2010/ParasR2MD_TestLSF
cd $MIRRWORKDIR
source  `which SetupProject.csh` Ganga v600r43 pytools
ganga $MIRRWORKDIR/Ganga_LSF/ganga.py &gt;&! $MIRRWORKDIR/Ganga_LSF/output.log
```

Where output.log should then store all of the output you would typically see on the screen, and of course you substitute **your** grid proxy location AND **your** workDir for `/afs/cern.ch/user/p/pnaik/RichAlignment/2010/ParasR2MD_TestLSF/`.

Also make a new simple python script `ganga.py` that has only the RichInfoName and execfile lines that you are used to entering interactively to run an alignment. For example the file would just have:

```
import os
print "renewing grid proxy (this will not ask for a password, if you renewed your grid proxy *bef
gridProxy.renew()
print "I am now going to give the command runMonitoring() [if we run this as a script and not fro
print "If the monitoring loop is already running, the runMonitoring() command will give an ERROR.
runMonitoring()
print "running the Offline RICH mirror alignment"
RichInfoName = ( os.getenv('MIRRWORKDIR') + os.path.sep + 'RichMirrorAlignmentGanga_Collision10_s
execfile( '/afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/job/Rich
```

where RichMirrorAlignmentGanga _Collision10_settings_R2MD.py is instead the name of your settings file

Then to you can submit the above `submit.csh` script to the batch queues, making sure of course that you write your grid proxy onto AFS:

```
chmod 755 submit.csh
lhcb-proxy-init -v 96:00
kinit -fp -r 7d
bsub -q2nd -n10 $PWD/submit.csh
```

The first line above changes the file to executable

The second line above creates a grid proxy valid for 96 hours

The third line creates an AFS ticket valid 7 days that is forwardable. We are not sure if it actually gets forwarded, but the ticket is valid for 7 days on the LSF machine (Paras checked)

The fourth line above submits the script to LSF. Note that we picked the 2 nominal days queue. This should give 2 days of CPU time which hopefully is enough to perform the 2012 alignments. -n10 means use 10 cores

It looks like you can't view any output until the whole job is done, but you can check your workDir to see if new iterations are being made every few hours or so. Also there are now Mirrors_*_touch.txt files that will show up while submitting grid jobs, while waiting for grid jobs, and when grid jobs are completed. They are empty, but if you `ls -al` them you will see the time stamp when they were created... if they aren't being created and updated fairly often (under every 30 minutes) during the Brunel stage of the alignment, then you know something is wrong with the alignment.

`bjobs` is the command to check your job status

`bqueues -u` gives the status of all the queues

`bkill` can be used to kill jobs if need be

You should get an email when your job has completed or failed.

Ancient Instructions

Working directory

▶ Show... ▼ Hide

- create a working directory (if it does not exist yet) and `cd` to it, e.g.:

```
mkdir -p ~/public/rich_align/workdir; cd ~/public/rich_align/workdir
```

- copy the following files to your working directory (assuming you are working with version `v3r3`):

```
cp $LHCBRELEASES/PANOPTES/PANOPTES_v4r1/Rich/RichMirrorAlignmentGanga/files/Rich2MirrCombinList_4
cp $LHCBRELEASES/PANOPTES/PANOPTES_v4r1/Rich/RichMirrorAlignmentGanga/files/Rich1MirrCombinList_1
cp $LHCBRELEASES/PANOPTES/PANOPTES_v4r1/Rich/RichMirrorAlignmentGanga/job/RichMirrorAlignmentGang
cp $LHCBRELEASES/PANOPTES/PANOPTES_v4r1/Rich/RichMirrorAlignmentGanga/scripts/getDataSet.py .
```

(nb some files will be added to the next release of `Panoptes v4r2` and may not be available in this current version. If the files are not available download them from the head version)

Compile the `Panoptes` projects in the usual LHCb way (cmt make).

Then set environments in your work directory

```
setenv HOME /afs/cern.ch/user/a/auser
setenv User_release_area $HOME/cmtuser
source `which SetupProject.csh`          =Panoptes v4r1=
source `which SetupProject.csh`          =LHCb=
rm                                    -f env_var_settings.txt
echo   $RICHMIRRORALIGNMENTGANGAROOT    &gt; env_var_settings.txt
```

Data (This example was written for Collision 11 Mag Up)

▶ Show... ▼ Hide

- Select data by executing the script getDataSet.py inside ganga and following the instructions then include the output file in the setting file below.

eventSelectorInputFile = "LFN-MagUp-R1.py"

brunelEvtMax = -1 brunelPrintFreq = 100 splitMaxFiles = 1000

splitFilesPerJob = 2

#backEnd = "LSF_1nh" backEnd = "dirac"

diracCPUTime = 19200

brunelInitialInputType = "DST" messageSvcOutputLevel = "3"

mirrCombinSubset = "48m85c_p12p43fix"

thisCase = "R2SelMd"

dataVariant = "Collision11" %ENDSYNTAX%

- ♦ some technical parameters specific to the "per-combination" fitting procedure:

```
groupSlices                      =  1
groupBins                        =  1
minAverageBinPop                 =  9
deltaThetaWindow                 =  4

coeffCalibrTilt                  =  0.3

#useOffsetsFromMC                = "true"
useOffsetsFromMC                 = "false"

#verOffsetsFromMC                = "129kMC10"
verOffsetsFromMC                 = ""

usePremisaligned                 = "false"

combinFitVariant                 = "slices"

maximumNumberOfIterations        =  8

startFromIteration               =  0

richDetector                     =  2 # 1: RICH1, 2: RICH2

magnifCoeffMode                  =  0 # 2: individual for each pair

solutionMethod                   =  1 # 1: algebraic, 2:Minuit
```

Running

▸ Show... ▾ Hide

- now start the engine:

```
In [2]:execfile( 'RichMirrorAlignmentGanga_Collision11_settings.py' )

In [2]:execfile( scriptOfEverything )
```

- and watch what `Ganga` says.

Scenarios

▸ Show... ▾ Hide
Sorry for the sparse population of this.

The key scenario is if the alignment breaks you can restart it by changing startFromIteration. This will re-run the whole iteration, which is safest. However if the Brunel jobs already were submitted and then the alignment was just wating for 80% of the jobs to come back you could skipFirstRecoJob. However, if you choose to do this then when you start ganga (always give the command "kinit -r 7d" before starting ganga, or if you forgot "!kinit -r 7d" within ganga, to renew your AFS token, and also of course make sure your grid proxy is still valid with lhcb-proxy-info or lhcb-proxy-info) you MUST wait until the monitoring loop starts again (i.e. printouts like "job 1123.51 changed to submitted"). Hit enter at the ganga prompt every couple minutes until you see evidence of this before starting the new alignment. Otherwise you may not get the updates needed for ganga to know that your jobs have finished.

We recommend copying the settings file and then using that one instead of the orginal one as the RichInfoName for your re-started alignment. Helps to keep track of stuff.

If for some reason after a while a restarted alignment with skipFirstRecoJob = 1 doesn't seem to be going anywhere (give it a chance, but i'd say a couple hours would be too long). Then just try restarting from whatever iteration with skipFirstRecoJob = 0.

You probably are not going to fill up all of your disk space, but if it is something you are worried about then just remove from ganga any jobs which are no longer needed (i.e. some failed alignment you never want to look at again), this will delete all local ROOT files as well associated with those jobs. (Remember, deleting a ganga job from ganga does not mean files associated with it are deleted from the grid [I haven't posted instructions on this yet, but for running alignments nothing is stored on the grid so don't worry about this]).

See the "Useful Ganga Commands" section of this twiki on how to remove jobs from ganga.

Pathway to RICH Mirror Alignment

The mirror alignment should use updated parameters from the Tracking corrections.

These talks willl help you to understand where RICH mirror alignment exists in the greater scope of HLT1 Alignment and Calibration:

- https://indico.cern.ch/event/320185/session/5/contribution/1/material/slides/0.pdf (Alignment/calib)
- https://indico.cern.ch/event/310501/contribution/2/material/slides/0.pdf (Alignment/calib)
- https://indico.cern.ch/event/310509/contribution/1/material/slides/0.pdf (OPG)

Return to the Top of this TWIki

Q & A

Is a refrative index calibration performed **before** the RICH mirror alignment?

- **UNKNOWN**... we should still be able to perform the mirror alignment... Sam knows how to do this...

Do we need the MC validation code anymore?

- PROBABLY NOT.... Was this only for initial tests of the procedure? Now that we know that it works this should not be necessary. (?)
- ASK ANATOLY

Return to the Top of this TWIki

Old Information, Postponed Indefinitely, or Obsolete

Current list of possible explanations for and differences in RICH1

▸ Show... ▾ Hide
Claire and Paras were discussing at LHCb-UK that it would be good to form a complete list of why the mirror alignment (resolution and/or constants) could/should depend on magnet polarity.

Something to keep in mind (from https://indico.in2p3.fr/event/1697/session/23/contribution/50/material/slides/0.pdf): The field strengths at the detector planes of each RICH are 1.4 mT (R1) & 0.4 mT (R2) So one could imagine magnetic field issues could be several times more prominent in R1 than R2 for this reason.

Here s what we can think of as possibilities (no matter how far-fetched):

* The mirror alignment itself introduces a bias

- Why it makes sense: If this occurred it would be the simplest explanation
- Why it doesn't: The mirror alignment knows nothing about magnet polarity as far as we can tell...

* The mirrors move

- Why it makes sense: obvious
- Question: What are the bolts made of?
- Why it doesn t: If the mirrors move then why (at least in the past) have we been able to use one set of constants for magUp and one for magDown. If the mirrors moved wouldn t these constants switch upon every polarity shift? Would the mirrors really just move back and forth between two discrete states?

* Our magnetic field is measured empirically since our magnet is a non-standard shape

- see http://lhcb-doc.web.cern.ch/lhcb-doc/presentations/ConferencePosters/Postcript/2007/Conti%20-%20CHEP07
- Why it makes sense: How often does this get updated? A small difference in accuracy of the magnetic field determination for magUp and magDown could carry through the reconstruction.
- Why it doesn t: Wouldn't we have observed problems anywhere else?

* MDMS corrections are not sufficient

- Why it makes sense: This system looks to study the effects of the magnet on the HPD detection of photons, so not getting this correct can cause an issue
- Why it doesn t: If these are measured and verified during data taking periods often enough then these should be accurate.

* There is something in the conditions database that indicates the magnet polarity; this propagates to the reconstruction

- Why it makes sense: it could happen, maybe we missed something
- Why it doesn t: Again, something like this should probably be noticeable elsewhere.

* Alignments before us in the chain change with polarity

- Why it makes sense: The mirror alignment depends on reconstructions which include alignments before us in the chain

- Why it doesn't: No known evidence of MagUp /MagDown differences in these alignments

We could be wrong about a fair bit of this. Any insight or thoughts on the above (or anything else to add to the list) would be useful...

Problems to be solved for past (i.e. 2012) RICH mirror alignments

▸ Show... ▾ Hide
(Task assigned to Paras, with help from Matt)

David Websdale (d.websdale@imperialNOSPAMPLEASE.ac.uk) found the 2012 alignment corrections appear odd, and (I think) he claims that there are only two distinct sets of corrections for the whole year, and also these sets differ by 1 mrad (whereas corrections in 2010 and 2011 fluctuate by less than 0.1mrad).

See https://indico.cern.ch/event/351162/contribution/3/0/material/slides/0.pdf

Problem is to be solved by using the conddb browser and compare the 2012 numbers in the file /afs/cern.ch/lhcb/software/DEV/DBASE/Det/SQLDDDB/db.bak/LHCBCOND_cond-20121008_.db.bz2 with the numbers in the current lhcbcond.db. Hopefully there will be a difference that will explain David's numbers

Update: There was no obvious difference. This is very odd and disturbing result. Matt said that the constants were updated fill by fill for 2012. Paras told the group that we will make sure that this is not a problem in 2015. Discussion on this topic has ceased after informing David and the other interested parties from the RICH. Paras is choosing not to follow up further as resources should be dedicated to making sure that this is not an issue in 2015.

2015: Question about necessity of separate Mag Up and Mag Down constants

▸ Show... ▾ Hide

- We will likely not be able to keep two sets of constants and switch in between for Run II, apparently
- There are improvements that have been made to the MDMS corrections that Chris Jones thinks might remove the need for separate constants for magUp and magDown
- The only way to test this is to (a) re-run the alignment on Run 1 data to see if this makes a difference. As far as we know, we don't have any person available to do this... Paras needs to focus on 2015. But if we have to do it we should find some way... (b) **IF** these improvements to MDMS corrections were made before Sam's 2010 reprocessing, then we could compare 2010 (post improvements) to 2011 (pre improvements) constant variations as a function of flip to see if there is a difference. Then maybe Claire (or anyone?) could look into it, as this would hopefully not take very long to study, as these constants are all there, and we believe Dave Websdale has done similar analysis of constants and may be able to help.

DATABASE SLICES FOR MDMS CORRECTIONS

Follow progress here:

https://its.cern.ch/jira/browse/LHCBCNDB-535

once released, just make sure you use the new DB tag, whatever that will be, for the Run1 Up/down studies.

Christian Linn is helping Jeremy and Paras perform these studies. The newest conddb and database tags are located here: http://lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/dbase/conddb/release_notes.xml
Tags in **red** are production release. Make sure the **year** is correct. Some tags do not work with some years.

For now we will use:

- For 2012 data
    - **cond-20150409-1** which includes the MDMS updates for 2012
    - **dddb-20150119-2**
- For 2011 data
    - **cond-20150409** which includes the MDMS updates for 2011
    - **dddb-20150119-1**

NOTE : We are not sure if the MDMS updates also need to be included in the dddb tags before we can proceed! (probably not, but if they do, this won't work)

The settings files for our 2012 tests will be kept in:

/afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/job/settings/tests/Run1MDMSU

(We have committed them to the HEAD version as well. Since these are just scripts you can pick them up from either place.)

So far we have only done this for IOV 04 Others should be very similar. We will make them if we have some time later, otherwise you should be able to figure out how based on the tables* below. This uses dddb-20150119-2 and cond-20150409-1 database tags, which have the Run 1 MDMS updates http://lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/dbase/conddb/release_notes.xml

Both Christian and Jeremy should do the following:

Adjust the settings file so that it will work in your directory structure.

To get the datafile:

SetupProject Ganga v600r43 pytools ganga ganga /afs/cern.ch/user/p/pnaik/cmtuser/Panoptes_HEAD/Rich/RichMirrorAlignmentGanga/scripts/getDataSet.py

Then (for data from IOV 04) enter the following at the prompts:

Enter year to align, 2011 or 2012?: 2012 Enter polarity to align, MagUp or MagDown?: MagDown Enter first run to use, zero for default (0): 114316 Enter last run to use, zero for default (9999999): 115464 OutputFile Name: MD_IOV04.py Enter Scale Factor (must enter an integer greater than or equal to 1): 1

For **other** IOVs please of course enter the appropriate information (polarity, run range, output file name) for that IOV

Run the alignment using the data sample produced as input (point to it in your settings file). We are now trying to run over all of the data in the IOV and see how long it takes with the new Brunel options. This is way more than we need, though it would be nice if it is fast, and it works. We ll see how far it gets overnight.

The number of events needed to align the RICH (RICH2 specifically) is historically 5 million. We think we can get away with 1.125 million with the improvements, so that is what we try now.

We are not sure if brunelEvtMax is the number of events Brunel runs per job or per subjob. We think it is the latter. If thats the case, we **think** one could go through just about 1.125 million events by changing these parameters in the settings file:

Christian, could you tell us what you used for these to get IOV 04 to converge?

splitMaxFiles = 750

splitFilesPerJob = 5

brunelEvtMax = 20000

So maybe Christian and Jeremy can try with this.

In any case, if there are enough events, the final iteration will output an .xml file with the mirror alignment.

Once this happens, we need to run the other IOVs (05, 06, 07, 11, 12, 15, 16) using the alignment of IOV 04 as input (i.e. we want to see if the alignment changes substantially). We haven't quite figured that process out yet, basically the easiest way would be to turn the .xml back into a .db slice, then add the .db to the AddedDBs of the other IOVs

[Hopefully this replaces the mirror alignment .db in the condDB before the .db gets read out again, have to check]

(Let me know if my assumptions about Brunel().EvtMax and about needing more than 5M events are possibly not correct)

Info needed for IOVs for 2012 include **these**...

code	calendar dates	run numbers	polarity	info
01	2012-04-01 / 2012-04-17	111183-112916	Down	resumed data-taking
02	2012-04-17 / 2012-05-01	113013-113146	Down	after VdM scan
03	2012-05-01 / 2012-05-02	114205-114287	Up	after technical stop, polarity change
04	2012-05-02 / 2012-05-16	**114316-115464**	**Down**	polarity change
05	2012-05-16 / 2012-05-31	**115518-117103**	**Up**	polarity change
06	2012-05-31 / 2012-06-11	**117192-118286**	**Down**	polarity change
07	2012-06-11 / 2012-07-02	**118326-118880**	**Up**	polarity change
08	2012-07-02 / 2012-07-20	119956-122520	Up	after technical stop
09	2012-07-20 / 2012-07-25	122540-123803	Up	after VdM scan
10	2012-07-25 / 2012-08-10	123910-125115	Down	polarity change
11	2012-08-10 / 2012-08-28	**125566-126680**	**Up**	polarity change
12	2012-08-28 / 2012-09-15	**126824-128268**	**Down**	polarity change
13	2012-09-15 / 2012-10-12	128411-129978	Up	polarity change, after technical stop
14	2012-10-12 / 2012-10-24	130316-130861	Down	polarity change
15	2012-10-24 / 2012-11-08	**130911-131940**	**Up**	polarity change
16	2012-11-08 / 2012-12-03	**131973-133587**	**Down**	polarity change
17	2012-12-03 / inf	133624-133785	Up	polarity change

... **AND** an example timestamp within each IOV

code	"date" timestamp
01	1333411200000000000
02	1334793600000000000
03	1335916800000000000
04	**1336089600000000000**
05	**1337299200000000000**
06	**1338595200000000000**
07	**1339545600000000000**
08	1341360000000000000

09	1342915200000000000
10	1343347200000000000
11	**1344729600000000000**
12	**1346284800000000000**
13	1347840000000000000
14	1350172800000000000
15	**1351209600000000000**
16	**1352505600000000000**
17	1354665600000000000

Rich Alignment person on shift at CERN (to be contacted 8-21h only)

2016

Show... Hide

- Shift DB (if you have to log in, then come back to this link again)

2015 Offline EM Magnet Down data

Show... Hide
Rich 1 10 GeV trigger track (default, for now): /afs/cern.ch/user/p/pnaik/ReduceRawFiles/EM/RICH1/10GeV/R1_10GeV_EM_Down.py 1936249 total triggers 1875361 events selected from 3580578 RICH1 and RICH2 combined line events

Rich 1 20 GeV trigger track (for studies): N/A

Rich 2 40 GeV trigger track (default): /afs/cern.ch/user/p/pnaik/ReduceRawFiles/EM/RICH2/40GeV/R2_40GeV_EM_Down.py 1787949 total triggers 1706643 events selected from 3580578 RICH1 and RICH2 combined line events

Implementation of Code in the Online Framework (Claire should give information for Rich/RichMirrorAlignmentOnline v0r3 and beyond in a separate section. Please preserve this section)

Show... Hide
Notes

- Paras has added and started work on the package `Rich/RichMirrorAlignmentOnline` within project `Panoptes`
- Claire now is the developer of `Rich/RichMirrorAlignmentOnline`.
- Much thanks to Beat and Roel for significant assistance
- Package `MirrAlignmentOnline` has already been deprecated. `OnlPanoptesSys` should be deprecated as well.
- Document describing the Framework is in the Attachments.
- Jordi Garra Ticó <jordi.garra.tico@cern.ch> is the **release manager** for Panoptes

Online Alignment Framework Sequence Information (based on Rich/RichMirrorAlignmentOnline v0r2. Claire should describe elsewhere the situation for v0r3 and beyond. Please preserve this for now)

Show... Hide

Hi Developers,

Paras has re-organized the famous RichMirrorAlignmentGanga.py driver script so that it can run both online and offline. This will be called RichMirrorAlignment.py for now, and placed in the same directory as RichMirrorAlignmentGanga.py for the time being. It will be committed soon (but is in my afs/cmtuser area [see the TWiki] if anyone wants to look). Currently both .py files are "identical" in functionality for Offline. Fairly soon we will want to transition to using RichMirrorAlignment.py exclusively; Paras has run the 2010 offline alignment with it, which is a good sign.

Benefits include only having one driver script to worry about if changes need to be made in the future. Downsides include having to be careful that anything we change has both Offline and Online in mind.

It's a major re-write of sorts, though the core of the software is the same. **Note that the ONLINE part is still incomplete and is being worked on** however the general behavior has been tested. The following is the general picture of what has/will happen with this script though:

There is no longer a while loop over the iterations until we reach the maximum number of iterations.

There is now instead an overall FSM while loop which is how the alignment will be run Online, but designed so that if we are not running Online it can run Offline/Ganga.

What happens now is that there are "major iterations" (iterationCount) and "minor iterations" (n_it). **In every minor iteration Online, Brunel MUST run**. This is because the Online framework is designed this way and it will not change.

In the new while loop, there are 10 minor iterations for every major iteration.

Each major iteration Offline: If minor iteration mod 10 is not 9, submit a job to Brunel via Ganga If minor iteration is 9, calculate the new alignment and if the alignment is done break out of the while loop, if not then increment the major iteration and continue the while loop. If major iteration > maximumNumberOfIterations break out of the loop

Each major iteration Online: If minor iteration mod 10 is not 9, run a Brunel job on the HLT farm If minor iteration is 9 calculate the new alignment and run a Brunel job on the HLT farm that we won't use If major iteration > maximumNumberOfIterations or if the alignment is done, tell the FSM that we are done

As you can see, Online we waste 1 out of every 10 Brunel farm CPU cycles, but the time wasted is "much less than the age of the universe" so we'll take it. Eventually we want to fix this somehow, but it can wait.

Online running involves a transition between various "states." This is detailed in the Framework for Online Alignment which you can find at the bottom of the TWiki. Regarding the Iterator (the new RichMirrorAlignment.py) the state transitions occur in a specific way. The following may seem a bit complex, but it is designed specifically so that the Analyzers that run Brunel on the HLT farm are properly synchronized with the Iterator which sets up Brunel and does all of the evaluation of results and production of new alignments. A Communicator coordinates the Iterator and the Analyzers.

The Online sequence for our alignment Iterator is:

Start Communicator

- set initial state: NOT_READY
- enter FSM while loop (**the FSM gives all commands** and can transition states, while **we can only transition states**)
- listen and wait for an FSM command

- first FSM command is "configure"
    - state transition: READY
    - continue while loop
- second FSM command is "start"
    - state transition: RUNNING
    - continue while loop
- third and subsequent FSM command until alignment is done is "pause" (this can **only** be given if the state is RUNNING)
    - state transition: PAUSED
    - collect histograms
    - do stuff that needs to be done ([A] give Brunel a tilt XML/db file OR [B] do MirrCombinFit and MirrAlign and come to a decision)
    - state transition :
        - RUNNING if we need to keep iterating, increment minor iteration if [A] or [B], increment major iteration only if [B]
        - READY if the alignment does not have enough events OR it has converged OR it reached the maximum number of major iterations
    - continue while loop
- penultimate FSM command "stop" (given when we go back to READY, but could be given while RUNNING)
    - state remains or transitions to READY
    - continue while loop
- final FSM command "reset"
    - set final state: NOT_READY
    - exit the FSM while loop

This is our new world, so study this, the FSM while loop is in RichMirrorAlignment.py so you can see what is going on when these FSM commands are received. One key thing is to realize that going back and forth between RUNNING and PAUSED allows us to run Brunel once each time in each xml configuration necessary (to be given between PAUSED and RUNNING), until the alignment is done.

Next steps: (i) Get the RichAnalyzer (already written) to work with the Communicator and Iterator. (ii) Get the RichAnalyzer to take in the .xml file with the conditions we want it to reconstruct the photons with (iii) Get the final .xml file to be written to our /group/online area in the correct way (iv) Figure out what information the monitoring needs to know and get that information to the monitoring (this includes plots of how well the RICHes are aligned, what the final status of the alignment was, etc..)

The plots should not be a difficult task but the monitoring will want things delivered in a certain way...

Once (i) is done then you should be able to run the test Online alignment with the instructions on the TWiki... you can try now just to understand how to run it, but for sure it will crash at the moment

P.S. None of the numbers used in this message have been hard coded. 🙂 9 is of course really len(tiltNames) and 10 is really len(tiltNames+1). This is of course to maintain compatibility with the old magnifCoeffModes, which have only a single tilt (i.e. untilted)

How to run the Online Mirror Alignment Tests for Rich/RichMirrorAlignmentOnline v0r2 (Claire should write this elsewhere for v0r3 and beyond. Please preserve this for now)

▸ Show... ▾ Hide

- Paras succesfully has imported a skeleton Online alignment into Panoptes `Rich/RichMirrorAlignmentOnline`, which is now being filled with pieces from the Ganga script.
- Instructions to run the current test

- - You need an Online account
    - Get one here: https://lbtwiki.cern.ch/bin/view/Online/UserFirstSteps
  - Always log in this way

```
ssh -Y lxplus.cern.ch
ssh -Y lbgw
ssh -Y plus
```

- - - recommend setting up password less logins, at least to lbgw from lxplus and plus from lbgw: https://lbtwiki.cern.ch/bin/view/Online/LoggingInFromUNIX (use Online username and password)
  - To get the code initially (replace "v5r5" with newest stable version, and **"v47r7"** with *the corresponding version of Brunel to that version of Panoptes, find it by clicking on "doxygen" in the appropriate row: http://lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/panoptes/releases/ *)

```
SetupProject --build-env Panoptes v5r5
getpack Rich/RichMirrorAlignmentOnline head
getpack PRConfig head
getpack Muon/MuonPIDChecker head
getpack Rich/RichMirrorAlignmentGanga head
getpack Rich/RichMirrAlign head
getpack Rich/RichMirrCombinFit head
getpack Rec/Brunel v47r7
```

- - - You can use getpack -p anonymous if you want to ignore all the repeated login requests. However the consequences of this are that you cannot commit. Best not to do this if you plan to edit the code here. Maybe best to edit the head on lxplus if you need to make improvements, then svn update as required.
  - There is one thing we needed to do... we are not sure if there is a better solution but **we had to edit the cmt/requirements file in PRConfig** to change:

```
set PRCONFIGOPTS $(PRConfig_root)/options

# Add the scripts to the python path
path_prepend PYTHONPATH "$(PRConfig_root)/python"
```

to

```
# Paras added this
use GaudiPolicy v*

#============================================================================
# Install python modules
#============================================================================
apply_pattern install_python_modules

# Paras commented these out
###set PRCONFIGOPTS $(PRConfig_root)/options

#### Add the scripts to the python path
###path_prepend PYTHONPATH "$(PRConfig_root)/python"
```

- - Use a new shell to build the packages

```
SetupProject --build-env Panoptes v5r5
cd $User_release_area/Panoptes_v5r5
make -j 8
```

- - Find a machine to run on

```
/group/online/dataflow/scripts/farmStatus &
```

-
    -
        - for example

```
ssh -Y hlte0902
```

-
    - To run the test

```
SetupProject Panoptes v5r5 --use PRConfig --use Rec/Brunel --use Muon/MuonPIDChecker
export PYTHONPATH=/home/raaij/pydim/lib/python2.7/site-packages:/scratch/jenkins/benchmark/python
cd $User_release_area/Panoptes_v5r5/PRConfig/scripts
python AlignOnlineTest.py --help
killall GaudiOnlineExe.exe
python AlignOnlineTest.py --numa --nodes=1 --workers=1 --task-log=log/PythonMirrAlignOnline.log -
```

Note that the directory (in this case /group/rich/2015-MirrorAlignData/20GeV/MagDown) must exist, and have RAW or MDF data files in it.

Once things are working increase `--nodes=1 --workers=1` to have more nodes and workers.

`killall GaudiOnlineExe.exe` should always be used before running the Online Alignment Tests

[Not sure about this yet:] A new temporary directory is created for each test and printed in the log that goes to the test log file and stdout. This directory should contain all intermediate files, results and conditions.

Coombes Automated Alignment Procedure

Matt's Coombes Automated Alignment Procedure page. This is probably postponed indefinitely, or obsolete.

Return to the Top of this TWIki

This topic: LHCb > LHCbRichMirrorAlignLegacy
Topic revision: r1 - 2017-07-21 - ParasNaik