MAF - Mimosa Analysis Framework Documentation

Auguste Besson, Damien Grandjean, Alexandre Shabetai
October 2003
Last update : June 20^{th} 2006

1 Introduction

This document contains a very short notice in order to perform an analysis of the test beam data through the MIMOSA Analysis Framework (MAF). Lists of available histograms are also given for reference.

This is NOT a tutorial. The code is not user friendly and not optimized since it has been developped successively by at least 6 overbooked physicists. If you don't know what a macro is actually doing, just check the code ! Please send comments or questions to :
abesson@in2p3.fr, damien.grandjean@ires.in2p3.fr, shabetai@ires.in2p3.fr.

2 MAF INSTALLATION

2.1 What is needed

- a tar file of the code :

  `maf\_VERSION.tar.gz`

- a gcc compiler, version $\geq$ 3.2
- a root version $\geq$ 4. (on SLC 3, please use root > 5.11)
- on Debian, a version $\geq$ 3.1.

2.2 How to install

- untar MAF :

  ```
  > tar -zxvf maf_VERSION.tar.gz
  ```

- have a look into MAF :

  ```
  > cd maf\_VERSION
  > ls
  Results  Scripts  bin  compiled  config  datDSF  datRAW  eta
  ```

  Results contains gif/eps files of results.
  Scripts contains configuration scripts.
  code contains the sources.
  bin contains the executable.
  config contains the data cards for each run and the eta distribution for the telescope planes.
  datDSF contains the reconstructed files.
  datRAW contains the raw data.
  eta contains eta distributions in root files for the tested detectors

- modify the script

  `Scripts/MAF-config`

  After the line 127, add a new configuration. For instance :

```
 # desy setup
sbgpcs78.in2p3.fr)
      # program path
      ROOTDIR=/rawcmos8/daq/MAF/maf
      # data path
      DTDIR=$ROOTDIR
      ROOTSYS=/rawcmos8/daq/MAF/root_slc
      CERNLIB=/cern/pro/lib
      # obj exec amd lib paths
      BUILD_PATH=/rawcmos8/daq/MAF/maf/bin
      ;;
```

The only things you should modify are :
sbgpcs78.in2p3.fr = domain name
ROOTDIR = working area
DTDIR = data path
ROOTSYS = root version
CERNLIB = cern lib path
BUILD_PATH = obj exec and lib paths

- WARNING. The command :

  ```
  /bin/hostname -f
  ```

  should gives you the complete domain name. If it is not the case, remove the "-f" option, line 111, and put a compatible domain name in your setup.

- you can now execute the MAF-config :

```
> source Scripts/MAF-config --nodebug

Configure the environnement for MIMOSA Tracking...

Setup for sbgpcs78.in2p3.fr

ROOTDIR    : /rawcmos8/daq/MAF/maf
DTDIR        : /rawcmos8/daq/MAF/maf
ROOTSYS      : /rawcmos8/daq/MAF/root_slc
LD_LIBRARY_PATH      : /rawcmos8/daq/MAF/root_slc/lib
:/rawcmos8/daq/MAF/maf/bin/Mimosa/LIB
BUILD_PATH=/rawcmos8/daq/MAF/maf/bin

Appending to PATH

Setup done!
Run make to compile or MAF to start working
```

- Check that your paths are correct. **You should execute** *MAF-config* **at the beginning of each session. To avoid this, you can execute it in your** *.profile* **or** *.bashrc***.**
- Compile the code :

```
> cd compiled
> make realclean
> make
```

- The executable should be here :

```
> ls $BUILD_PATH/bin/MAF/BIN/
MAF
```

3 How to get raw data and link them

- To get raw data, connect the Analysis PC (sbgpcs26) to the DAQ PC.
- To transfer data, create one directory per run, (for instance */rawcmos8/data/mimosa8_desy_raw_092005_dir01/8510*).
- sftp daq, login : daq, password : OR, sftp rootdaq, password :

```
> cd /data2/desy_mi8_09_2005_2/8510
> mget *
```

- Raw data can be anywhere in your disks. The script *Scripts/renameRawData.sh* is devoted to create the correct links between the actual place of raw data and the directory *datRAW*. In *renameRawData.sh*, you have to modify :
  #—-RUN NUMBER
  Run_Number=512
  Mimosa_Number=8 *Run number will be 8512*
  #—-FILES NUMBERS
  BEGIN=0 *Start index for the raw data file (example : 510_000.raw)*
  END=11 *number of files*

  #—-DIRECTORY WHERE THE links will be created :
  Raw_data_Path=/rawcmos8/daq/MAF/maf/ *working directory*
  Raw_data_Path2=datRAW
  #—-DIRECTORY WHERE THE RAW DATA ARE :
  Raw_data_Path3=/rawcmos8/data/mimosa8_desy_raw_092005_dir01

- to execute the script, just do :

```
> Scripts/renameRawData.sh
```

4 How to setup the data cards

- The data cards can be found here :
  config/runXXXX.cfg .
- Plane 1-8 correspond to the 8 telescopes planes. Planes 9 and 10 correspond to the 2 matrices of the tested chip.
- Here are presented only variables which you may need to modify in a typical configuration.

// Run Parameter
RunNumber : 8512 *Run number*
EventsInFile : 10000 *Events per file*
StartIndex : 0 *Start index for the raw data file (example : 510_000.raw)*
EndIndex : 20 *Number of files*

// Parameters of the Tracker
Planes : 10 *Total number of planes*
TracksMaximum : 1 *keep at 1*
HitsInPlaneMaximum : 1000 *Maximum number of hits recorded by planes*
Resolution : 3. *track resolution in microns, see chi2 distribution of tracks*

// -*- Reference Plane (1-8)
InitialPedestal : 400 *number of events used to compute the Pedestal*
InitialNoise : 400 *number of events used to compute the Noise*

Tilt1 : -0.32 *Alignment angle given by the tracker alignement*
AlignementU : -43.678 *alignment U given by the tracker alignement*
ThreshNeighbourSN : 3.0 *S/N of 8 neighbours Threshold for clusters*
ThreshSeedSN : 7. *S/N Threshold seed for cluster*
Status : 2 *1 = primary reference*
(i.e. fixed plane for alignment ; 2 = secondary reference (i.e. mobile plane for alignment)
// -*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*- MIMOSA Plane (9-10)
ModuleType : 2 *keep at 2*
ModuleNumber : 1
InputNumber : 1
ChannelNumber : 1 OR 513 *put at 1 for the first MIMOSA plane and at Channels+1 for the second mimosa plane*
Channels : 512 *number of channel i.e. read pixels*
StripselUse : 0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF 0xFFFF Name : "MI-MOSA" Purpose : "" Readout : 5 *CDS onboard = 7, CDS offline = 5, MIMOSA 9 special plane = 8*
AnalysisMode : 2 *keep at 2*
InitialPedestal : 400 *number of events used to compute the Pedestal*
InitialNoise : 400 *number of events used to compute the Noise*
StripsU : 32 *number of pixel in U direction*
StripsV : 16 *number of pixel in V direction*
PitchU : 0.040 *pitch in mm*
PitchV : 0.040 *pitch in mm*
PitchW : 0. *keep at 0*
StripSizeU : 0.040 *pitch in mm*
V : 0.040 *pitch in mm*
ThreshNeighbourSN : 2.0 *S/N of 8 neighbours Threshold for clusters*
ThreshSeedSN : 5.0 *S/N Threshold seef for cluster*
ClusterLimitU : 0.080 *around 2 times the pitch*
V : 0.080 *around 2 times the pitch*
W : 0. *keep at 0*

// Parameter of the Data Acquisition
// WARNING max size of buffer currently 0x10000
// you may change that in include/DAcq
// exact number of Bytes in the FileHeader
// the fileheader is written once per file
//FileHeaderSize 1024
FileHeaderSize : 0
// exact number of Bytes in the EventBlock
// this is the length of a raw event
// 2 planes * 32 * 16 * 32 bits / 8 bytes + 116 + 8 planes * 256 * 8 bits/8 bytes
EventBufferSize : 6260
0x1874 *same number in hexadecimal (put 0x in front of it)*
// here : 74 stands for the header, 8 stands for the 8 ref Planes, 1 for mimosa. // EventBufferSize is different for data of run Oct'99 and June'00
// non critical number of characters for printing the event
// header information, say about 80
FileHeaderLine[d] : 0
// The number of Module types used, e.g. Ref planes, Mimosa, etc.
AcqModuleTypes : 2

// -*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*- Sirocco-A *reference planes*
Name : "Sirocco-A"
Devices : 1
Inputs : 4
Channels : 2048 *total number of channels in the 8 reference planes (256x8)*
Bits : 8 *number of bits per channel*
SignificantBits : 8
// -*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*- Sirocco-B *Mimosa planes*
Name : "Sirocco-B"
Devices : 1
Inputs : 1
Channels : 1024 *total number of pixels in the chip (plane 9 + 10), including dummy pixels*
Bits : 32 *number of bits per pixel*
SignificantBits : 32

5 Data reconstruction

- Raw data need to be reconstructed. The reconstructed file is like *datDSF/run8510_01.root* . If for any reason, you want to reconstruct the data again (with different cuts for instance), the new file will be named *datDSF/run8510_02.root* and so on. By default the Analysis (see next section) will be done on the latest reconstructed file.
- The reconstruction is divided in 3 phases, Eta generation for telescope planes, Tracker alignment, and reconstruction itself.

5.1 Eta generation for telescope planes

- Run MAF :

```
> MAF
```

- It will open a root session. All the available methods can be called with the command :

```
root [1]gMAF->NameOfTheMethod(parameters)
```

- Before each method, you need to Initialise, i.e., read the data card.

```
root [1]gMAF->InitSession(8510)  //(run number)
root [2]gMAF->MakeEta(10000)
(number of events used to generate Eta functions)
```

- The result is stored in *config/eta_8510.root*.

5.2 Tracker alignement

- Now align the tracker :

```
root [1]gMAF->InitSession(8510)
root [2]gMAF->AlignTracker(700)  //(bound distance in micron for the fit)
```

- The result is stored in data cards *config/runXXXX.cfg*.
- If alignment crashes the first time, run it again (this bug will be corrected soon).

5.3 Reconstruction

- Run the reconstruction :

```
root [1]gMAF->InitSession(8510)
root [2]gMAF->DSFProduction(300000)  //(number of event to process)
```

- The reco files are in the directory *datDSF/* . It is a preselection of hits and tracks based on the cuts you applied in the data card.

6 Analysis

6.1 Setup of hard coded parameters

- For technical reasons, some important parameters are hard coded in the source. Here is a list of parameters you may want to change. Of course, you need to compile again if you modify such parameters.
- PixelSize, NofPixelInRaw, NofPixelInColumn, calibration (in e-/ADC), initial parameters for chip alignement (degres and microns) : in MPara.cxx (GetParameters(Option_t* Option)).
- Window size for the tracks-hits association which corresponds to GeoMatrix values in MimosaPro method. (in MPara.cxx (TrackInMimo)). **See help section for more details**.

6.2 Processing the analysis

- The first thing to do is to align now the tested chip reference frame to the telescope reference frame (see next session for more details).
- As mentioned above, the file *runXXXX_xx.root* can now be processed trough *MimosaPro* method :

```
root [1] gMAF->InitSession(8510,10) //(run number,plane number)
root [2] gMAF->MimosaPro(300000,100,4,2,2,3,"no",0)
```

- By default, for a given run, the last reconstructed file will be used. If for any reason, you want to select another reconstructed file, use the *SetFileNumber* method. For instance, if you have reconstructed the Run 14532 three times, you will have three reconstructed files in the *datDSF* directory :

```
run14532\_01.root  run14532\_02.root   run14532\_03.root
```

  By default *MimosaPro* will process *run14532_03.root* . If you want to process *run14532_02.root* for instance do :

```
root [1] gMAF->InitSession(14532,10)
root [2] gMAF->SetFileNumber(2)
root [3] gMAF->MimosaPro(300000,100,4,2,2,3,"no",0)
```

- Here are the parameters of MimosaPro :
  - Int_t MaxEvt = number of events to process
  - Int_t TrackHitDist = Cut on the distance between Hit position and track extrapolation position (microns). The Track-hit distance need to be lower than this cut to consider a track associated to a hit.
  - Float_t S2N_seed = Signal to Noise cut
  - Float_t S2N_neighbour = Signal to Noise of the 8 neighbours cut
  - Int_t submatrix = Submatrix number (1 corresponds to S1, etc. see MPara.cxx section)
  - Int_t GeoMatrix = Window for tracks (0,1,2,3. 3 corresponds to the submatrix with the 1 column/line removed on the edge. see MPara.cxx
  - Option_t* SaveAlign = If *yes*, MimosaPro will save the generated eta functions for the mimosa plane and will open a Minuit session to aligne the tested chip.

- Int_t UseHitMap = 0 or 1. (0 = DOT NOT consider Hit map cut. 1 = consider Hit map cut(=read or store the map)).
  To remove hot pixels from analysis, you need first to generate a hot pixel map in a first processing. Then, in a second processing, you can read this map to remove bad pixels. Before running MimosaPro, chose between generate and read the map :

  ```
  root [1]gMAF->SettHitMapReadOpt(1) (0=store, 1= read)(default at 1).
  ```

  The threshold of the number of hits leading to a Hot pixel is defined in MimosaAnalysis.cxx, line 286 :

  ```
  // if the pixel is a seed too many times in the run, remove the hit.
    Float_t CUT_MaxNumberOfHitsPerPixel = 800;
  // you can also remove pixels with low occupancy for testing:
    Float_t CUT_MinNumberOfHitsPerPixel = 0;
  ```

6.3 Mimosa alignment

6.3.1 Preliminary remarks

Before alignment, there are 2 reference frames which are not necessarly equals :

- The (HU,HV) reference frame is related to the plane under study (mimosa). It is fixed. The U and V directions in the mimosa chip are defined by the way pixels are read and stored in the raw data. Data are usually stored line by line. These lines define the U direction. (see figure 1). The origin (0,0) of the frame is the center of the matrix.
- The (TU,TV) reference frame is related to the track extrapolation.

The goal of alignment is to determine the shifts (U,V) and the 3 possible rotation angle (th1, th2, th3) so that the 2 reference frames are superimposed.

- Since the readout of each mimosa plane is not necessayly done in a same direction, one needs to do one alignment per plane under study.
- If no alignment has been done, which should be the case the first time you do it, the default parameters (trU,trV in microns and theta[0],theta[1],theta[2] in degres) are taken in the file MPara.cxx. For instance :

```
if((RunNumber >= 14000) && (RunNumber < 15000)
  ){
  RunExists=kTRUE;
  MaxNofPixelsInCluster = 25 ;
  NofPixelInRaw = 66;
  NofPixelInColumn = 128;
  PixelSize = 30.;
  Matrixtype=11;
  if(ThePlaneNumber == 9){ //MAT 1 RAD TOL
    //---alignement
    theta[0]= 180.0;
    theta[1]= 180.0;
    theta[2]= 0.0;
    trU= 900.0;
    trV= -430.0;
    //---
  }
  else if(ThePlaneNumber == 10){  //MAT 2 STANDARD
    //---alignement
    trU= -1010.0;
```

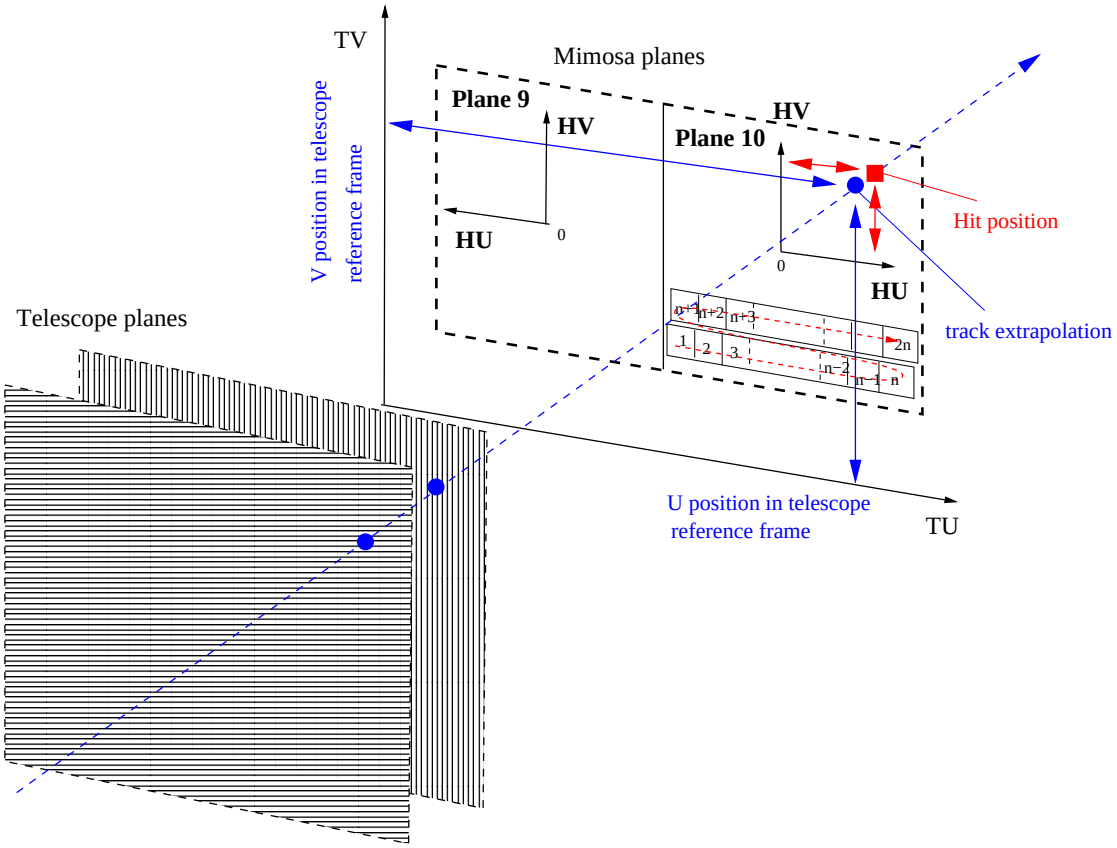


FIG. 1 – Mimosa Alignement : Readout example and corresponding reference frames

```
    trV= -440.0;
    theta[0]=180.0;
    theta[1]=180.;
    theta[2]= 0.0;
    //---
}
```

- Please note that you could copy the file eta/CorParXXXX_9.root from a previous run. Though be aware that the alignment should be done for each run if you want to get a satisfactory residual.
- By default, plots are in micron units.
- Run first MimosaPro with the options TrackHitDist = 80000, GeoMatrix = 0 and SaveAlign = "no".
- then check the default alignment with the *Check alignment* method in the menu. It shows correlations between Hit position in the chip frame and hit position in the telescope frame. If the alignement is correct, the correlation is close to a line like $f(x) = x$ (top plots of figure 2). You can also see the residuals (track position - Hit digital position). The digital residual should be centered in zero (bottom plots of figure 2).
- Now click on *Check Ref.Tracks* (or *gMAF->Checkreferencetracks()*). On Canvas 1, you can see the figure 3 (if the alignment is already correct), representing all tracks which are not associated to a hit in mimosa. It means that there are no hit closer than the *TrackHitDist* cut. The trigger used has roughly a $2 \times 4\mu m^2$ dimension. The white square corresponds to the shadow of the analysed matrix. This plot shows that it will be necessary to remove all the tracks not going trough the detector under test. This will be done thanks to the *geomatrix* option which sets minimal and maximal value in U and V of the tracks. This plot is also

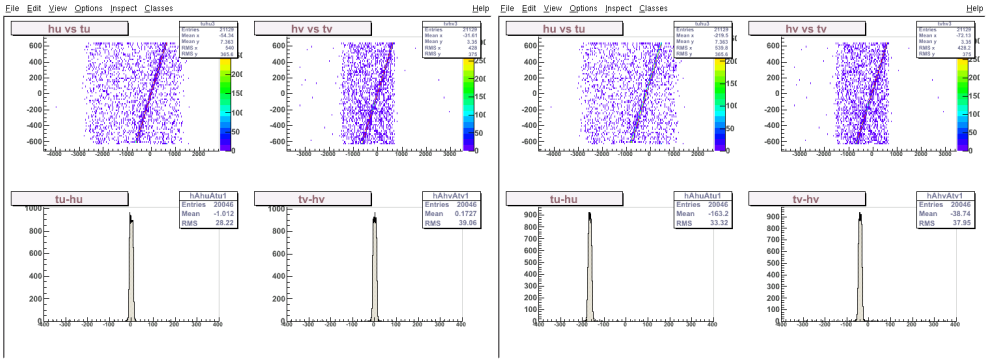


FIG. 2 – Check Mimosa Alignement (good/bad)

very useful to check that the trigger and the detector under test are correctly aligned.

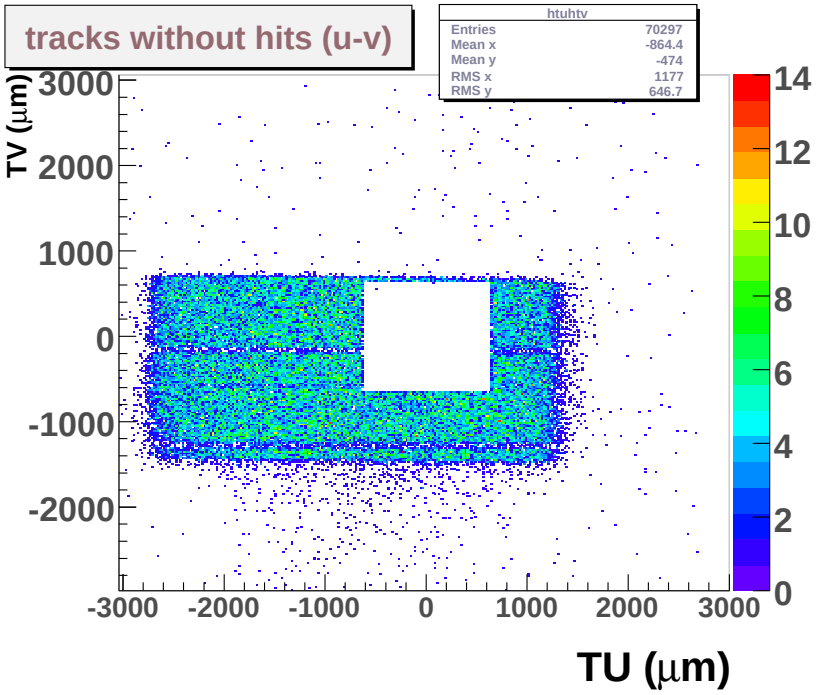


FIG. 3 – Tracks not associated to a hit in the (TU,TV) reference frame (with a good alignment)

- As a matter of fact, to align the tested detector, you can either set the option *SaveAlign* to *"yes"* when you run MimosaPro or run the command *gMAF->AlignMimosa(1)* :

```
root [1] gMAF->InitSession(14532,10)
root [2] gMAF->MimosaPro(300000,10000,4,2,2,0,"yes",0)
```

OR

```
root [1] gMAF->InitSession(14532,10)
root [2] gMAF->MimosaPro(300000,10000,4,2,2,0,"no",0)
root [3] gMAF->AlignMimosa(1)
```

which will open a *Minuit* session. The 5 parameters of the fit are U,V shifts and the 3 rotation angles th0, th1 and th2. The chip alignement is a recursive procedure. You must do it in several steps. To begin, you should set a large *TrackHitDist* cut, and the largest window

(GeoMatrix = 0). Then step by step, you will reduce these cuts. With low energy beam, a good χ^2/NoF should be below $\simeq 100$ (Since the errors coming from bad associations and multiple scattering are not taken into account, the χ^2 is usually far above 1).

- To help you the correlation plots between U,V position and residuals DU and DV are shown in microns (see figures 4 to 8). You should get no correlation on all these plots (i.e. an horizontal distribution). a shift of this line indicates a bad X or Y parameter whereas a non horizontal line indicates a bad angle th0 or th1 (th2 is always near 0). On the left plots, when you get a residual like a gaussian centered on zero, it means that you are close to the right alignement.
- The typical minuit command you may need :
  - set,par,1,3.14 = set the parameter number 1 to 3.14 value (angles must be given in radians, shifts in microns).
  - mini = minimize χ^2
  - fix,par,1 = fix the parameter number 1 to its present value.
  - exit = to quit Minuit. The program will ask you if you want to save the alignment you have obtained :

    ```
    Store the results of alignement? (1/0)
    ```

    answer 1 to save, 0 to ignore.
- Sometimes, Minuit won't manage to find a good alignment alone. It can happen if the minimization is realised with many tracks outside the chip and many hits not correlated to a track. You'll have to guess the rough parameters. Try to set different combinations of angles (usually par1 = 0 or π, par2 = 0 or π, par3 = 0) with the commands *set* and *fix*. Once you get horizontal lines, it means that you got roughly the good angles. Now set the shifts U and V. Exit and save this alignement without minimizing anything with minuit.
- Please refer to Minuit documentation for more details.

  `http://wwwasdoc.web.cern.ch/wwwasdoc/minuit/minmain.html`

6.3.2 Examples

```
root [1] gMAF->InitSession(14532,10)
root [2] gMAF->SetFileNumber(2)
root [3] gMAF->MimosaPro(150000,500,4,2,2,3,"yes",0)

 **********
 **   43 **CALI           8
 **********
 Theta1= -180.402406
 Theta2= 0.021322
 Theta3= 0.013318
 Shift U= 541.480947
 Shift V= 629.201700
 Z= 0.000000
 N of entries 20046 Chi2 189122983.078848 Chi2/DoF 9434.449919
command : -180.402406
retour code : 3

   ==>List of MINUIT Interactive commands:
 CLEar     Reset all parameter names and values undefined
 CONtour   Make contour map of the user function
 EXIT      Exit from Interactive Minuit
```

```
FIX        Cause parameter(s) to remain constant
HESse      Calculate the Hessian or error matrix.
IMPROVE    Search for a new minimum around current minimum
MIGrad     Minimize by the method of Migrad
MINImize   MIGRAD + SIMPLEX method if Migrad fails
MINOs      Exact (non-linear) parameter error analysis
MNContour  Calculate one MINOS function contour
PARameter  Define or redefine new parameters and values
RELease    Make previously FIXed parameters variable again
REStore    Release last parameter fixed
SAVe       Save current parameter values on a file
SCAn       Scan the user function by varying parameters
SEEk       Minimize by the method of Monte Carlo
SET        Set various MINUIT constants or conditions
SHOw       Show values of current constants or conditions
SIMplex    Minimize by the method of Simplex
```

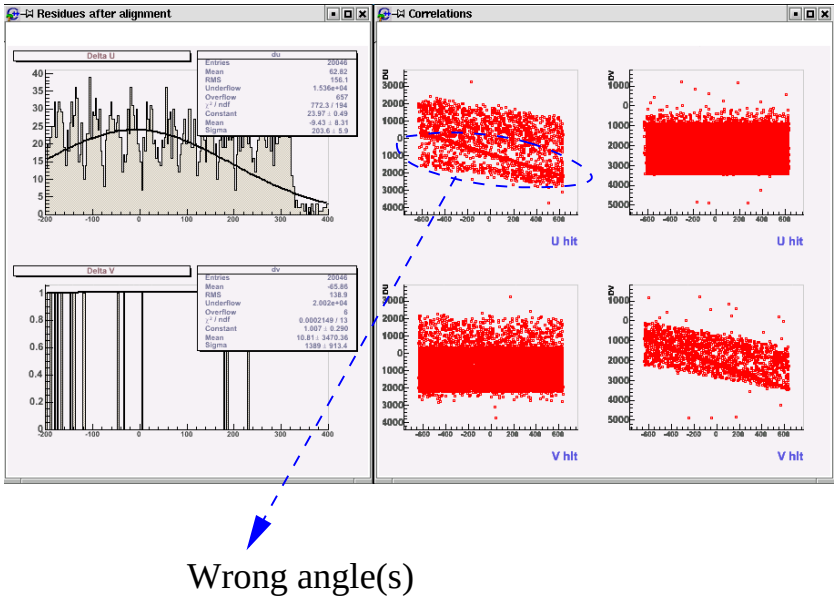


FIG. 4 – Mimosa Alignement (1) : Bad alignement

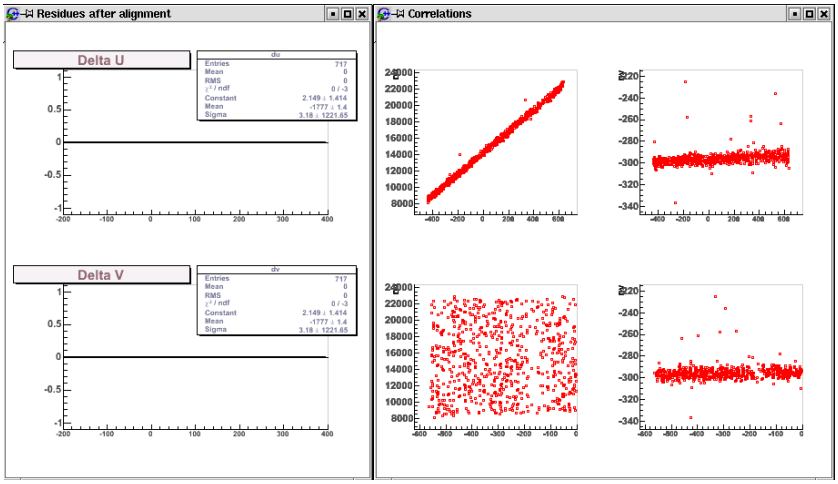
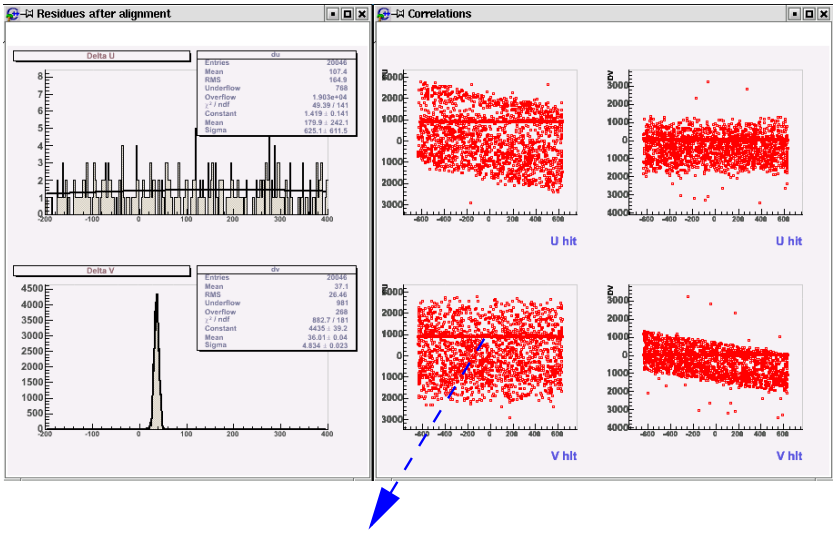


FIG. 5 – Mimosa Alignement (1) : Only one angle is good



Residual is shifted by ~900 microns.
You must increase par4 by ~900 microns.

FIG. 6 – Mimosa Alignement (2) : Good angles but bad shift

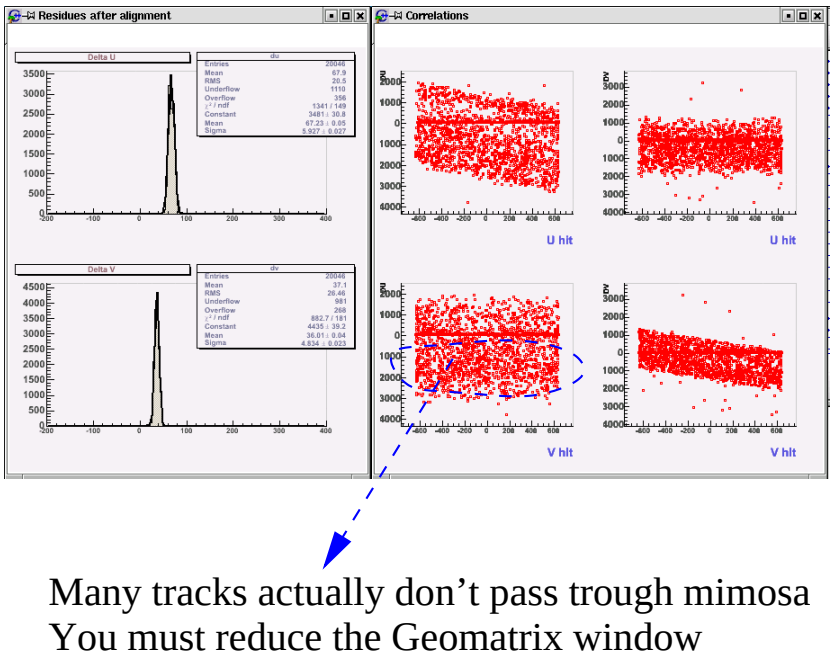


FIG. 7 – Mimosa Alignement (3) : Almost good alignement but bad window for tracks

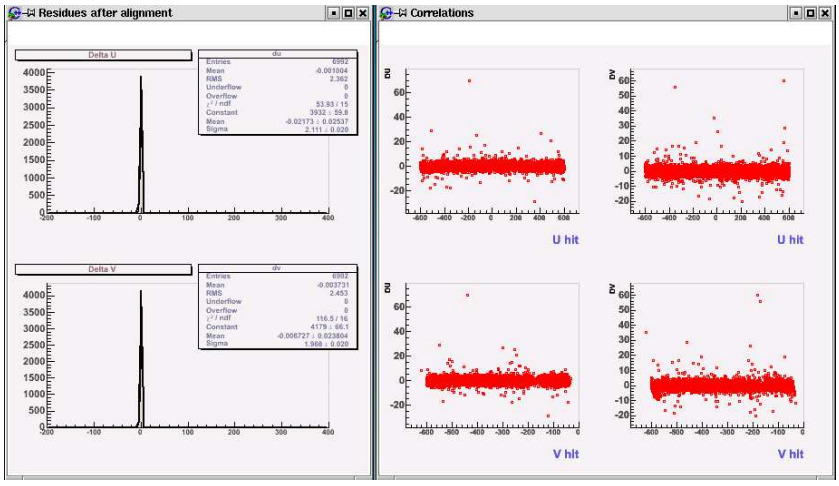


FIG. 8 – Mimosa Alignement (4) : Good alignement

6.4 Eta generation for the tested detector

- To generate the eta function of the tested detector you can either :

  - set the option *SaveAlign* to *"yes"* when you run MimosaPro,

  - or use the method :

  ```
  root [1]gMAF->StoreEta()
  ```

  just after MimosaPro. That will store the result in *eta/Cor#.root* files.

- Please keep in mind that you need to run again MimosaPro if you want the eta distribution to be taken into account.

6.5 Plot results

- The menu shown in figure 9 allows to run the different macros.

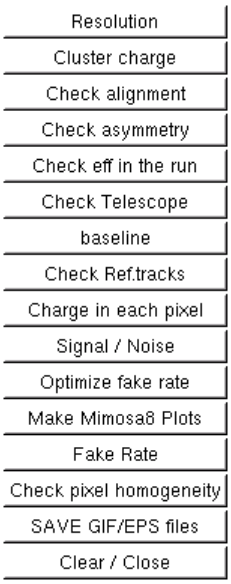


FIG. 9 – MimosaPro menu

- Once MimosaPro has run, you can also plot any histogram, without calling any macros, by using the pointer of each histogram (see the list below). For instance :

  ```
  root [1]gMAF->CDHist()
  root [2]HistogramPointer->Draw()
  ```

- Resolution

  ```
  root [1]gMAF->MimosaResolution()
  ```

  - *hEtaURes* : Residual between track extrapolation and hit position determined by eta 3x3 function (U direction)

  - *hEtaVRes* : Residual between track extrapolation and hit position determined by eta 3x3 function (V direction)

  - Residual for different methods : Center of Gravity (U and V) ; eta2x2 (U and V) ; eta3x3 (U and V)

- Cluster charge

  ```
  root [1]gMAF->MimosaClusterCharge()
  ```

  - *hqcn[0]* : Charge in seed (e-)

- *hqcn[8]* : Total Charge of the 9 highest charge pixels among a 5x5 cluster (e-)
- *hqcn[24]* : Total Charge of the 5x5 cluster (e-)
- *hRealTrackNoise* : Noise of the seed pixels (e-)
- *hsnc[0]* : Highest Signal/Noise of the cluster (usually the seed pixel)
- *ChargeSpread* : MPV value of the charge vs the number of pixels included in the cluster. WARNING : the 25 pixels are ordered by decreasing charge.

• Check alignment

```
root [1]gMAF->CheckMimosaAlign()
```

- *hAllHuvsAllTu* : hUdigital vs tu, i.e. Seed digital U position vs U track position for all hits
- *hAllHvvsAllTv* : hVdigital vs tv, i.e. Seed digital V position vs V track position for all hits
- *hAllHuvsAllTu1* : tu-hUdigital, i.e. (U track position - Seed digital U position) namely digital residual for all hits
- *hAllHvvsAllTv1* : tv-hVdigital, i.e. (V track position - Seed digital V position) namely digital residual for all hits

• Check asymmetry

```
root [1]gMAF->CheckAsymmetry()
```

- *OBSOLETE*

• Check eff in the run

```
root [1]gMAF->InspectEfficiency()
```

- *greff* : Efficiency for each set of 1000 events vs event number.
- *grnum/grevt* : Good hits and good tracks vs event number.

• Check Telescope

```
root [1]gMAF->checkRefPlane()
```

- Method to control the 8 reference planes device. For expert only.

• baseline

```
root [1]gMAF->CompareBadGoodRaw()
```

- One arbitrary pixel is taken as reference. To check that the baseline is stable.
- *hraw1goodone* : raw value on this pixel before CDS for good events (i.e. correct track-hit association)
- *hraw1badone* : raw value on this pixel before CDS for bad events (i.e. no track-hit association)

• Check Ref.tracks

```
root [1]gMAF->Checkreferencetracks()
```

- Different useful plots for analysis.

* **Canvas 1** :

- *htuhtv* : U and V positions of all tracks selected by the *GeoMatrix* windows (tu,tv) which are NOT associated to a hit. See *MimosaPro(...,...,...,...,...,GeoMatrix,...,...)*.
- *htu* : U position of all tracks selected by the *GeoMatrix* windows (tu) which are NOT associated to a hit.

- *htv* : V position of all tracks selected by the *GeoMatrix* windows (tv) which are NOT associated to a hit.
- *allhits* : U vs V digital position of hits passing the *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* cuts AND NOT necessarly associated to a track. (hUdigital,hVdigital) (hUdigital,hVdigital)
- *hqcn[16]* : Total Charge of the 17 highest charge pixels among a 5x5 cluster (e-)
- *goodhits* : U vs V digital position of hits passing the *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* cuts AND associated to a track. (hUdigital,hVdigital)
- *vec* : U vs V digital residual of all hits passing the *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* cuts AND associated to a track. (tu-hUdigital,tv-hVdigital)
- *DuvCG* : Distance between track extrapolation and the closest hit (microns). Hit position is determined with Center of gravity method.

* **Canvas 2** :

- *hAllS2N* : Signal/Noise of all reconstructed hits (not neccessarly associated to a track). For these hits, the cuts of the runXXXX.cfg file (ThreshNeighbourSN,ThreshSeedSN) are applied but the cuts in *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* are NOT applied.
- *hallhitSN* : same but zoomed in
- *hPedestal* : DUMMY histogram
- *hnGOODhit* : Number of good hits (cuts in *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* are applied) in each event
- *hgoodSNneighbour* : S/N of the 8 highest pixels among the 24 neighbours. (sum of charge over quadratic sum of noise).
- *hnahitievt* : Number of good hits in mimosa vs event number (10000 events are represented). (cuts in *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)* are applied)
- *hsnn[0]* : Highest Signal/Noise of the cluster (usually the seed pixel) ? ? ? ? ? ? ? ? ? ?
- *hnhit* : Number of hits in all planes (including ref planes). ? ? ? ? ? ? ? ? ? ?
- *hnhitievt* : Number of hits vs event number ? ? ? ? ? ? ? ? ? ?

* **Canvas 3** :

- *h2DSeedPixel* : Digital position of seed pixel inside the matrix (in pixel units)
- *hdistchi2* : Distance between track exptrapolation and hit vs Chi2/NoF of tracks which could be associated to a hit
- *hqcn[3]* : Total Charge of the 4 highest charge pixels among a 5x5 cluster (e-)
- *hchi2* : Chi2/NoF of all tracks
- *hchi2_c* : Chi2/NoF of tracks which could not be associated to a hit
- *hchi2_nc* : Chi2/NoF of tracks which could be associated to a hit
- *hSNReal* : S/N(seed)
- *hSNNReal* : S/N(8 neighbours)
- *hSN_vs_SNNReal* : S/N(seed) vs S/N(8 neighbours)

• Charge in each pixel

```
root [1]gMAF->pixelcharge()
```

- *OBSOLETE*

• Signal / Noise

```
root [1]gMAF->MimosaSN()
```

- *OBSOLETE*

- Optimize fake rate

```
root [1]gMAF->MimosaOptimize()
```

  - theoritical fake rate for a given set of cuts. It assumes that noise is perfectly gaussian (which is not usually true).

- Make Mimosa8 Plots

```
root [1]gMAF->Mimosa8plots()
```

  - Analysis for digital Pixels (no noise and pedestal calculation)

- Fake Rate

```
root [1]gMAF->FakeRate()
```

  - Plot experimental fake rate (Valid for GeoMatrix=5). You have to set a Geomatrix where all reconstructed tracks are actually not going into the chip under test. In this case, if we assume that the beam flux is low enough, there should not be real hits in the chip for these events. Since this assumption is not completly true, this algorithm can only provide a very conservative estimate of the fake rate. Of course, a better way to do this estimate would be to use runs when the beam is off.
  - The fake rate is defined as the probability for one pixel to have a fake hit (i.e. coming from noise fluctuation) for a given set of cuts provided by *MimosaPro(...,...,S2N_seed,S2N_neighbour,...,...,...,...)*.

- Check pixel homogeneity

```
root [1]gMAF->MimosaPixelHomogeneity()
```

  - Eta performances and resolution studies. Only useful if the telescope resolution is good enough (i.e. if multiple scattering is negligible).

  * **Canvas 1** :

  - *hHOM_tu_tv_modulo* :
  - *hHOM_ResU_tu* :
  - *hHOM_ResV_tv* :
  - *ProfHOM_ResU_tu* :
  - *ProfHOM_ResV_tv* :
  - *hHOM_Charge_diodedist* :
  - *ProfHOM_Charge_diodedist* :
  - *hHOM_SNseed_diodedist* :
  - *ProfHOM_SNseed_diodedist* :

  * **Canvas 2** :

  - *hEta3U* :
  - *hEta3V* :
  - *hHOM_DU_Nevent* :
  - *hHOM_DV_Nevent* :
  - *ProfHOM_Charge2_diodedist* :
  - *hHOM_modUeta3_modUCG* :
  - *hHOM_modVeta3_modVCG* :
  - *rofHOM_Charge25_diodedist* :

- *hHOM_Charge25_diodedist* :

* **Canvas 3** :

- *hHOM_modtu_Nevent* :
- *hHOM_modtv_Nevent* :
- *hHOM_modUCG_Nevent* :
- *hHOM_modVCG_Nevent* :
- *hHOM_Charge_diodedist_40_50* :
- *hHOM_Charge_diodedist_50_60* :
- *hHOM_Charge_diodedist_60_70* :
- *hHOM_Charge_diodedist_70_80* :
- *grCharge_diodedist* :

* **Canvas 4** :

- *hHOM_modUeta3_modtu* :
- *hHOM_modVeta3_modtv* :
- *hHOM_modUeta3_realtu* :
- *hHOM_modVeta3_realtv* :
- *hHOM_modUCG_realtu* :
- *hHOM_modUeta3_Eta3U* :
- *hHOM_modVeta3_Eta3V* :
- *hHOM_modUeta3_modVeta3* :
- *hHOM_modUCG_modVCG* :

- SAVE GIF/EPS files

```
root [1]gMAF->SaveGifFile()
```

  - It will save in eps and gif format all the canvas created by the used methods.
  - Results are stored in the directory :

    `Results/14532_pl9_Sub1_Seed5_Neig2_geomat3_T2h_HitMap0`

    The values correspond to respectively Run number, planenumber, Submatrix number, S/N cut on seed, S/N cut on neighbours, Track to hit distance cut, Use of hot hit map or not.

- Clear / Close"

```
root [1]gMAF->Clear()
```

  - Clear everything to start over

7 Help

- To get the list of the main methods, just do :

```
root [1]]gMAF->Help()
++++++++++++++++++++++++++++++++++++++++++++++++++
       COMMONLY USED COMMANDS
++++++++++++++++++++++++++++++++++++++++++++++++++
gMAF->Clear()
gMAF->Help()
```

```
gMAF->CDHist()
-----------------------------
------------2/ RECONSTRUCTION
-----------------------------
--->Initialise Session (RUN,PLANE 9/10)
gMAF->InitSession(,)
--->Make eta function for telescope tracker (NEVENT)
gMAF->MakeEta()
--->Align tracker (DISTANCE CUT primary planes - secondary plane(microns) )
gMAF->AlignTracker(700)
--->Reconstruction (NEVENT)
gMAF->DSFProduction(150000)
-----------------------------
------------3/ ANALYSIS
-----------------------------
gMAF->SetfCUT_MaxNumberOfHitsPerPixel(2000)
--->Analyse reconstructed data
(NEvt,TrackHitDist,S2N_seed,S2N_neighbour,
 submatrix,tracker window,SaveAlign,hitmap)
gMAF->MimosaPro(10000,200,5,2,1,0,"yes",0)
gMAF->MimosaProM8(...)
gMAF->Mimosa8plots()
gMAF->MimosaResolution()
gMAF->MimosaClusterCharge()
gMAF->CheckMimosaAlign()
gMAF->CheckAsymmetry()
gMAF->InspectEfficiency()
gMAF->CompareBadGoodRaw()
gMAF->MimosaSN()
gMAF->InspectEfficiency()
gMAF->CheckAsymmetry()
gMAF->InspectEfficiency()
gMAF->checkRefPlane()
gMAF->CompareBadGoodRaw()
gMAF->Checkreferencetracks()
gMAF->pixelcharge()
gMAF->MimosaSN()
gMAF->MimosaOptimize()
gMAF->StoreEta()
gMAF->MimosaPixelHomogeneity()
gMAF->AlignMimosa()
gMAF->SaveGifFile()
```

7.1 How to

- Each time you modify a code don't forget to compile again ! (in case you declare a new histogram, you may even need to do a *make realclean*).
- modify the chi2/Nof cut on track ? Modify its cut in the *MPara.cxx* file :

```
TrackChi2Limit          = 3.;
```

- Modify Nevent Range For Efficiency vs time plot ? Modify in the *MAnalysis.cxx* file :

```
Int_t NeventRangeForEfficiency = 1000;
```

- use a baseline cut (avoid bad reset effects on mimosa 5) ?

  - Modify in the *MAnalysis.cxx* file :

```
Bool_t usebaselinecut = kFALSE ;
```

  - and set the variable to *kTRUE*

- For debugging you might want to print out informations in an ASCII file.

  - Modify in the *MAnalysis.cxx* file :

```
Option_Store_Events_in_file==1
```

[...]

```
if(Option_Store_Events_in_file==1){
 outFileEvt<<"####### evt RealEventNumber "<<ievt<<endl;
}
```

- Add a new histogram ?

  - Declare it in *MHist.cxx* and *MHist.h* :

```
h123  = new TH2F("h123","title",160,-20,20,160,-20,20);
```

[...]

```
h123 = Zero(h123);
```

[...]

```
TH2F *h123;
```

  - Fill it *MAnalysis.cxx* :

```
h123->Fill(x,y);
```

  - Draw it in the macro : (*MPost.cxx*) :

```
AutoZoom(h123)->Draw("colz");
```

- Create a new method ?

  - Declare it in *MAnalysis.h* :

```
  void MimosaPixel();
```

  - Write it in *MPost.h* :

```
void MimosaAnalysis::MimosaPixel()
{
[...]
}
```

  - Add a new button in the menu in *MPost.h* :

```
bar2->AddButton("button name","gMAF->MimosaPixel()","comments");
```

- Setup calibration factors ?

  - Edit *MPara.cxx* in the corresponding area :

```
if((RunNumber >= 14000) && (RunNumber < 15000)
  ){
[...]
 if((RunNumber == 14500)&&(ThePlaneNumber == 9 )){ calibration =  8.12;}
[...]
}
```

- Setup geomatrix window ?

  - Edit *MPara.cxx* in the corresponding area :

```
case 14: //MimosaType
    switch(ThePlaneNumber){
    case 9: // ThePlaneNumber // RAD HARD MATRIX
      switch(GeoMatrix){
      case 0: trkumin=-10000.;trkumax=10000.;
        trkvmin=-10000.;trkvmax=10000.;break;
      case 1: trkumin=-1500.;trkumax=1500.;
        trkvmin=-1500.;trkvmax=1500.;break;
      case 2: trkumin=-1200.;trkumax=1200.;
        trkvmin=-1500.;trkvmax=800.;break;
      case 3: trkumin=-900.;trkumax=870.;
        trkvmin=-1850.;trkvmax=-500.;
      if((RunNumber==14547) ){trkumin=-900.;trkumax=870.;
        trkvmin=-1850.;trkvmax=-500.; }
      break;
      default:;
      }//end of WindowType
      break;
    case 10: // ThePlaneNumber //STANDARD MATRIX
      switch(GeoMatrix){
      case 0: trkumin=-10000.;trkumax=10000.;
        trkvmin=-10000.;trkvmax=10000.;break;
      case 1: trkumin=-1500.;trkumax=1500.;
        trkvmin=-1500.;trkvmax=1500.;break;
      case 2: trkumin=-1000.;trkumax=1000.;
        trkvmin=-1400.;trkvmax=720.;break;
      case 3: trkumin=-900.;trkumax=900.;
        trkvmin=-1300.;trkvmax=630.;break;
      default:;
      }//end of WindowType
      break;
    default:;
    }
    if (tv<trkvmax && tv>trkvmin && tu<trkumax && tu>trkumin)
        TrkCrossMimo=kTRUE;
    break;
```

- the cases 0,1,2,3 correspond to the geomatrix option in *MimosaPro*.

- Modify read out if raw data format has changed ?

  - Modifying the data format needs to be done in *DPlane.cxx* in the *Int_t DPlane : :Update()* method.

- Zoom automatically on an histogram ?

```
AutoZoom(h123)->Draw("colz");
```

- Reduce the minimal number of ref planes to accept a track ?

  - Edit *DTracker.cxx* in the *void DTracker : :find_tracks()* method

```
Int_t    requiredHits(7); //default = 7
```

8 Checking raw data event by event (not updated ! ! !)

You can check the raw data event by event without processing it. It can be useful to check the data quality. For instance :

```
> cd $ROOTDIR
> MAF
root [1] gMAF->InitSession(9552,9)
root [2] gMAF->GetRaw()
root [3] gMAF->GetRaw()->InitScan()
root [4] gMAF->GetRaw()->MimosaDisplay()
root [5] gMAF->GetRaw()->MimosaDisplay()
root [6] gMAF->GetRaw()->MimosaDisplay()
root [7] gMAF->GetRaw()->InspectScan()
root [8] gMAF->GetRaw()->MimosaDisplay()
root [9] gMAF->GetRaw()->MimosaJump()
```

9 Notes (experts only)

- **WARNING** : in the MimosaPro source file, you can chose which hit position algorithm is chosen to do the alignement (digital, center of gravity, Eta functions). The function call is like :
  alignement.NewData(UofHitEta3,VofHitEta3,tx,ty,tz,tdx,tdy)
- configuration network on sbgpcs26 : more /etc/network/interfaces
- Align TRACKER once it is done the inf#.root file seems to be not usable anymore.
- Check Reference detectors init session + init scan + RS display.
- MIMOSA 9 : has a special feature. Matrices 64 x 64 and 32 x 32 recorded in the same time. So while one frame is read on the first submatrix, 4 frames are read on the second submatrix. This has to be taken into account mainly in *source/DPlane.cxx, mi9_mat_a3.c, event_header.typ* .

  ```
  Put readout = 8 in the config file.
  ```
- If one reference plane is broken. Then you have to reduce the minimum number of hits to build a track.

  void DTracker : :find_tracks()
- check data format : uncomment these lines in DPlanes.cxx Update.

  ```
  /*
  if((fPlaneNumber==9)||(fPlaneNumber==10)){
  FprintEventHeader (DAS_VGEvent.PtHeader, 1);
  printf("----end of event = %x \n",*DAS_VGEvent.PtEOR);
  //     getchar();
  }*/
  ```
- dans intiscan.C :

  ```
  tSession->SetEvents(5000); // < 5000 1600
  ```

  This number has to be greater than the number of events you want to scan.
- DEBUG MODE IN DPLANE look for 'test.dat print' InitScan.C

  ```
  //test.dat print
  //fprintf(test,"\n -------EVENT = %d --------- \n",i+1);


  //DPlane.cxx:
  int fopen_lockfile(){
    static int vFirstcall = 1;
  ```

```
  if(vFirstcall==1){
    VG_Lockfile= fopen("test.dat","a");
    printf("fopen_lockfile  is called \n");
    vFirstcall=0;
  }
}
.... + les prints.

//DTracker.cxx:
  extern  int fopen_lockfile();
extern  FILE * VG_Lockfile;

//DTracker.cxx:
DTracker::DTracker(DSetup& c, DAcq& aAcq){
  fopen_lockfile();

//PAREIL dans DTrack.cxx

// test.dat print
//  fprintf(VG_Lockfile,"test");

// test.dat print
// fprintf(VG_Lockfile,"PIXEL = %d ; frame 1 = %d ; frame 2 = %d \n",
stp,tmpRawFrame1,tmpRawFrame2);

// test.dat print
// fprintf(VG_Lockfile,"\n ------------------- \n");
```

- Tracker update.

  call find_tracks

  passer le parametre = nombre minimum de plan pour reconstruire une trace. A definir en variable globale dans init session ?

```
../macros/new/AlignTracker.C:    tTracker->Update();
../macros/new/AlignTracker.C:    tTracker->Update();
../macros/new/InitScan.C:  tTracker->Update();
../macros/new/MAalignTracker.C:    tTracker->Update();
../macros/new/MAalignTracker.C:    tTracker->Update();
../macros/new/MakeEta.C:    tTracker->Update();
../macros/new/MimosaDisplay.C:  tTracker->Update();
../macros/new/MimosaDisplayFunc.C:  tTracker->Update();
../macros/new/RSDisplay.C:    tTracker->Update();
../macros/new/ScanBaseline.C:  tTracker->Update();
../macros/new/ScanBaseline.C:  tTracker->Update();
```

- DPlane : :CalculateCommonMode() REMOVE :

```
if(fPlaneNumber>2){
  printf(" DPlane, Plane %d, CommonMode Error in Region %d,
  investigate !!!\n",fPlaneNumber,region+1);
}
```

- Tracker alignement (1) :

  In the config file, you set the status of the 8 references planes. You must have 4 primary

reference plane for alignement and 4 secondary planes. In case you have 2 planes broken, set the planes in such a way that you still have 4 primary planes.

- Tracker alignement (2) : Sometimes the fit doesn't converge correctly. You can play on several parameters (very empirical) :

```
AlignTracker.C -> const Float_t tiniBound = 1000.0 ;
AlignTracker.C -> tSession->SetEvents(5000)
AlignTracker.C -> Float_t tiltZold[4]={180.,90.,0.,90.};
AlignTracker.C -> Float_t Big=1000.;
AlignTracker.C ->   counts+=1000;
config file -> change primary and secondary planes.
```

- If you use USB comment these lines in DReader.cxx

```
for(ViBuff=0;ViBuff<(VSize32);ViBuff++){
   VPtBuff32[ViBuff]=R32_F(VPtBuff32[ViBuff]);
}
```

10 SPS BEAM CONTROL SYSTEM

- Login to the PC :
  user : eax7a ; passwd : spsC1WX7A
- Run the system :
  clic on shortcut then clic on "START"
- Beam status :
  Delay, DWC5 and DWC6
- TRIM 3 = vertical ; TRIM 4 = horizontal. 40 A / cm.
- DOOR Control :
  clic on access, chose the right door.

11 X-rays irradiation at CERN

- Building 13-14, 5th floor, room 5-004.
- Contact person : Federico.faccio@cern.ch (6th floor)
- Rule for the dose :
  For 40kV and 3 cm distance, we have the equation :
  Rad/min = 120 + 455.69 x I(mA)
  Examples :
  10 mA corresponds to 4.68 kRad/min, so 20 kRad in 4 min 17s.
  5 mA corresponds to 2.34 kRad/min, so 20 kRad in 8 min 33s.
- Distance to set the apparatus :

  1/ for instance, we want to put the apparatus at 3 cm of the chip, and this chip is at 11 mm of the hole of your box. Uniformity should be ok on a 15 mm diameter circle.
  2/ take a reference height. Put the apparatus on contact to the plastic box. Read the height at the top right.
  3/ increase the height to +19 mm.
  4/ check with the diode (swith on with the box in the bottom left, set the intensity, ans put the mirror).
  5/ REMOVE THE MIRROR AFTER YOU CHECKED THE ACCEPTANCE ! ! ! ! ! (otherwise you will only irradiate the mirror)

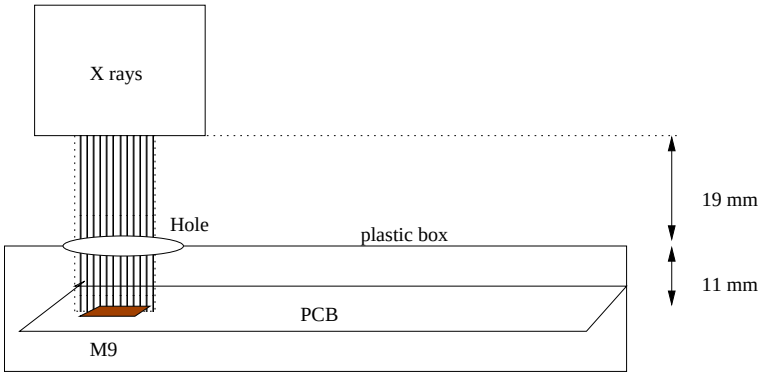


FIG. 10 – X ray setup

- Alim Chip. 7.5-8 V :

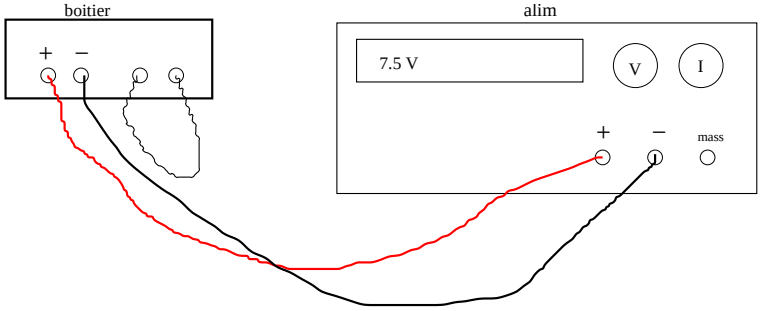


FIG. 11 – Chip alim

1/ Set V 7.5 - 8 V. Adjust the Current if necessary.
2/ plug the alim.

- Key put to “ON”
- Chose the power manually : 40 kV, 5mA, enter.
- Timer Active : (symbol “clock 1”). put the watch, set the time, enter.
- Swith ON : button “START”
- OPEN X-RAYS : button “1” (F4)
- PAUSE = 1

12 Useful numbers

- Gatignon phone number : 75 391 OR 16 35 34
- SPS control room : 77500
- X7A : 00 41 22 76 76058