Measurements of UVCI instrumental polarization

Capobianco G., Fineschi S.

Report nr. 115 date: 23/12/2008

INDEX

1. Introduction

Starting from 4 images $m_i (i = 0,1,2,3)$ acquired at 4 different angles, where usually rotation angle are:

$$\begin{cases} i = 0 \rightarrow 0\,\text{deg} \\ i = 1 \rightarrow 45\,\text{deg} \\ i = 2 \rightarrow 90\,\text{deg} \\ i = 3 \rightarrow 135\,\text{deg} \end{cases} \quad (1)$$

We can write:

$$m_i = \frac{1}{2}\sum_{j=0}^{2} a_{ij} \cdot s_j \quad (2)$$

Where s_j are the Stokes parameters and a_{ij} are first row elements of the Mueller matrix.

In literature:

$$\begin{cases} s_0 = I \\ s_1 = Q \\ s_2 = U \end{cases} \quad (3)$$

Stokes parameters are 4. $s_3 = V$ is the parameter for circular polarized light not including in this report cause KPol polarimeter work with linear polarized light.

$$\begin{cases} a_{i0} = 1 \\ a_{i1} = \cos(2\vartheta_i) \\ a_{i2} = -\sin(2\vartheta_i) \end{cases} \quad (4)$$

where ϑ_i are the rotation angles. From KPol polarimeter calibration these values are:

$$\begin{cases} \vartheta_0 = -0.8\,\text{deg} \\ \vartheta_1 = -50.2\,\text{deg} \\ \vartheta_2 = -89.8\,\text{deg} \\ \vartheta_3 = -136.2\,\text{deg} \end{cases} \quad (5)$$

Inverting eq. 2, we obtain values for $s_0, s_1 \; and \; s_2$.

Finally, polarized brightness (pB) is:

$$pB = \sqrt{s_1^2 + s_2^2} \quad (6)$$

KPol use nematic liquid crystals, than rotation angles are settings applying different voltages to LC. Voltages V_i for angle reported in (5) are [1]:

$$\begin{cases} V_0 = 10000mV \\ V_1 = 4150mV \\ V_2 = 3050mV \\ V_3 = 2450mV \end{cases} \qquad (7)$$

Resuming, a array is the follow:

$$a = \begin{pmatrix} 1.00000 & 0.999610 & 0.027922 \\ 1.00000 & -0.180519 & 0.983571 \\ 1.00000 & -0.999976 & 0.006981 \\ 1.00000 & 0.041876 & -0.999123 \end{pmatrix} \qquad (8)$$

Inverting this array we found a^{-1} :

$$a^{-1} = \begin{pmatrix} 0.267627 & 0.246785 & 0.233534 & 0.252055 \\ 0.516075 & -0.023557 & -0.480393 & -0.012124 \\ 0.063648 & 0.495361 & -0.047230 & -0.511779 \end{pmatrix} \qquad (9)$$

Screenshot of Mathematica notebook for this operation is in fig. 1.1.

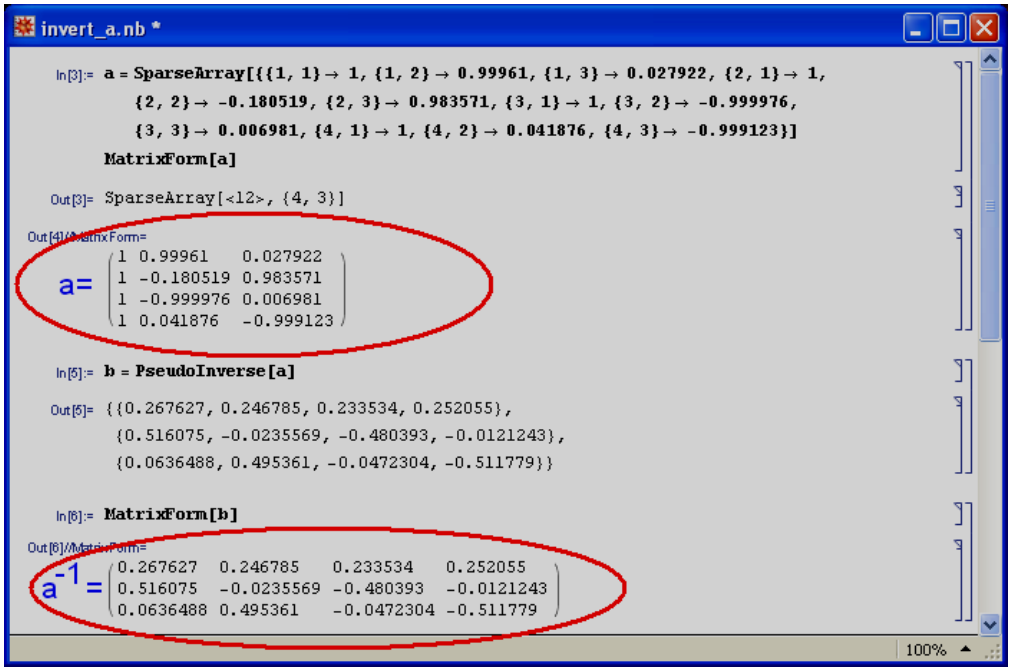

Figure 1.1 - *Inversion of a array*

2. Instrumental polarization

2.1 Set-up

During this tests, HERSCHEL was in scotch tank in air (fig.2.1). Light source was a Newport lamp with the F/10 NRL diffuser. A pinhole of 10.0 mm (solar diameter) and a 2ND filter were also used . A pre-polarizer was mounted in front of the telescope mirrors (behind the entrance pupil on M0) for the first sequence data acquisition (Set-up #1) and in front of the KPol for the last calibration (Set-up #2) (fig. 2.2).

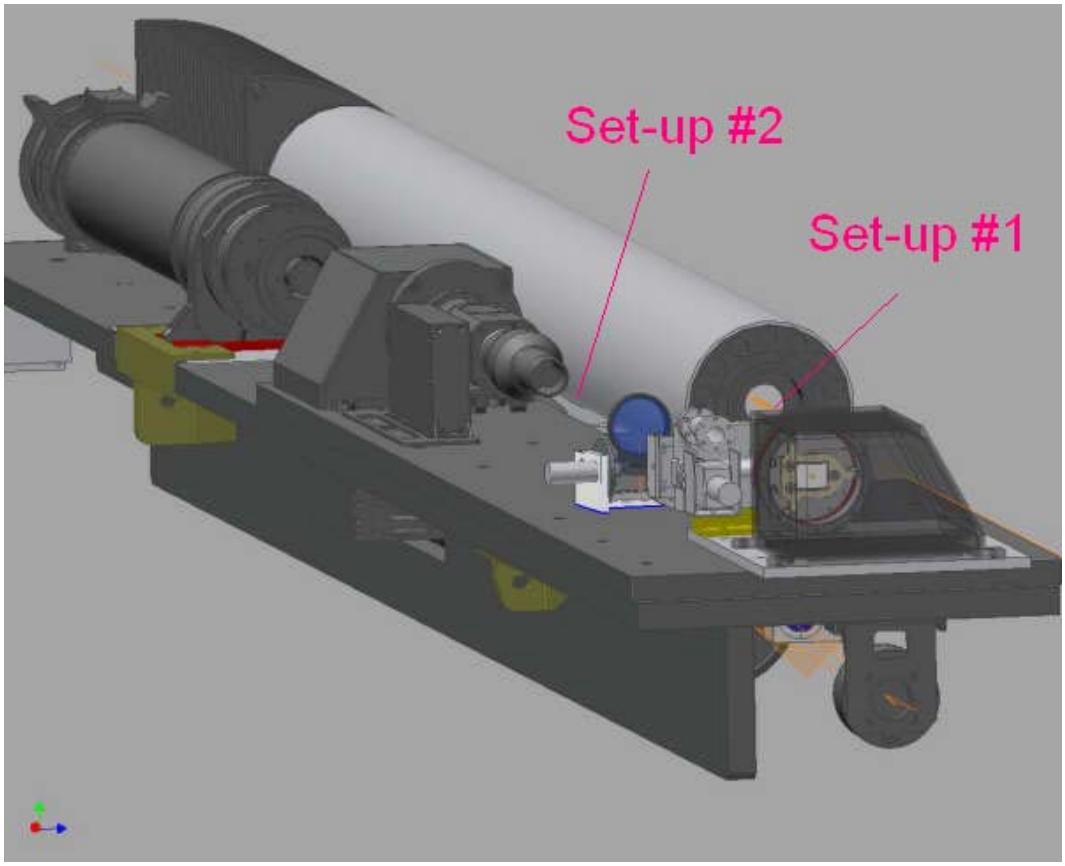

Figure 2.2 – *Schematic Assembly with position of pre-polarizer*

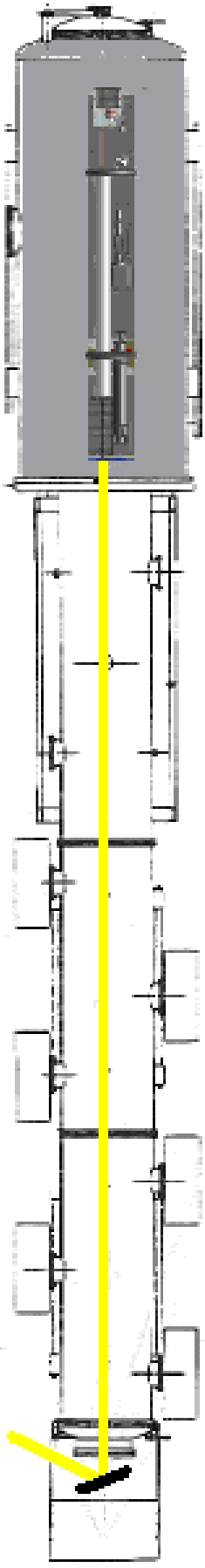

Figure 2.1 – *Schematic view of HERSCHEL in NRL tank*

2.2 Data Analysis

A way for evaluate UVCI instrumental pB is using two sets of measurement. The first one obtained with a pre-polarizer mounted in front of M0 mirror, than before telescope optics (M1 and M2) and a second set obtained with pre-polarizer mounted in front of KPol (after M1 and M2). A schematic view of telescope mount is in fig. 2.3.
A picture of instrument after integration is in fig. 2.4. Difference between pBs of this two sets give us instrumental pB.

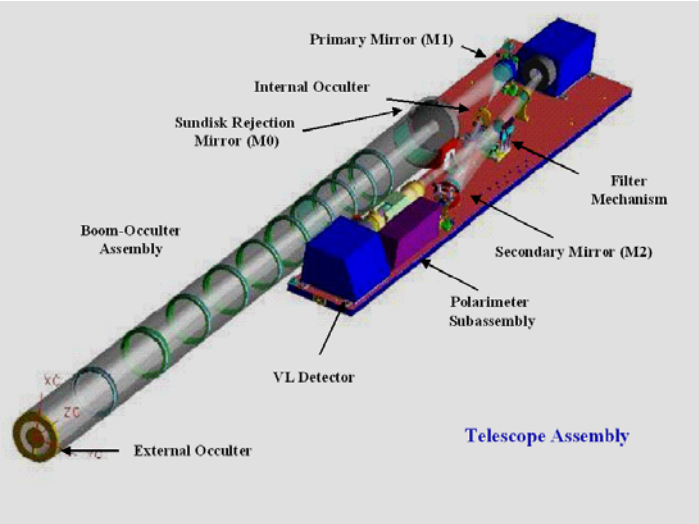

Figure 2.3 – *Schematic Telescope Assembly*

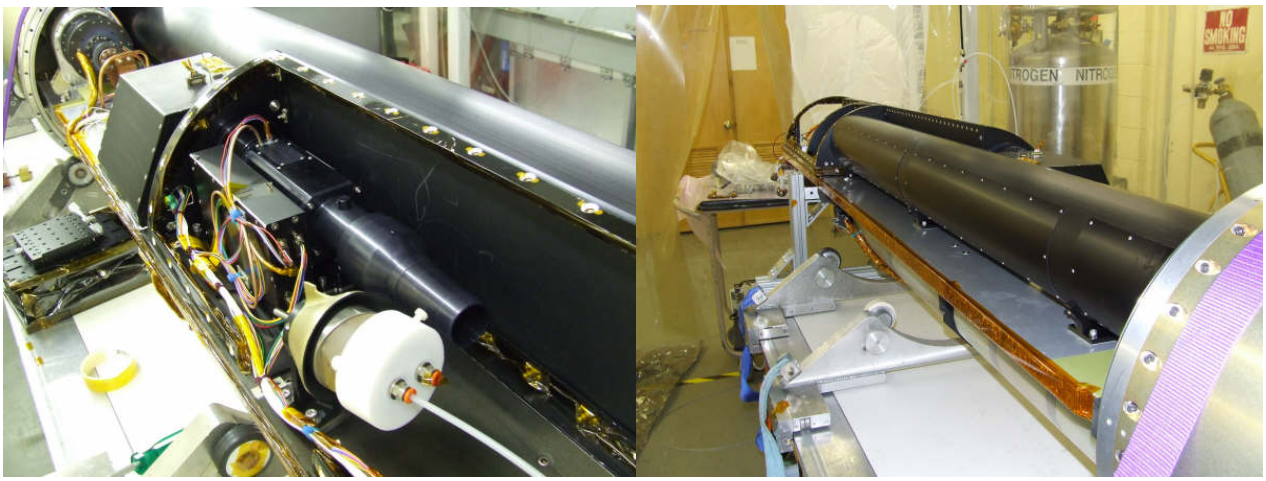

Figure 2.4 – *UVCI Instrument (Polarimeter and boom details)*

Data used are listed in Appendix A.

For data acquired mounting pre-polarizer in front of M0 [set-up 1], we measure:

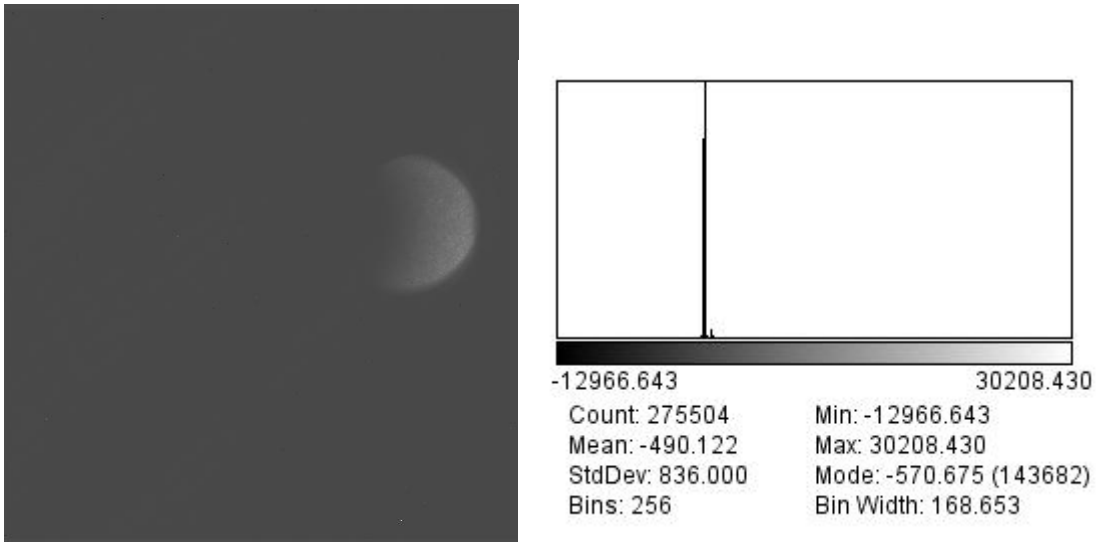

Figure 2.5 – *[Set-up #1] I parameter*

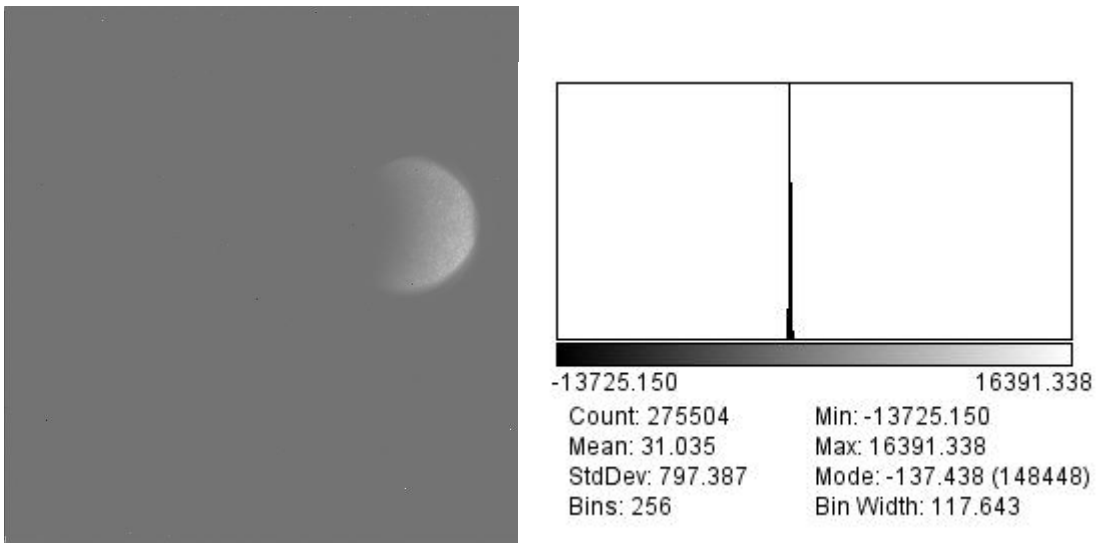

Figure 2.6 – *[Set-up #1] Q parameter*

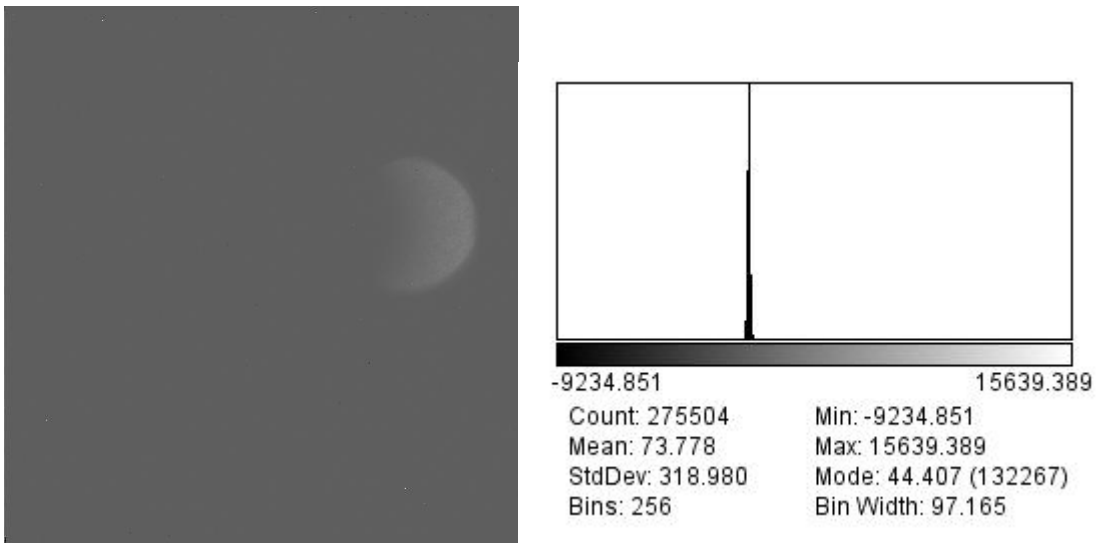

Figure 2.7 – *[Set-up #1] U parameter*

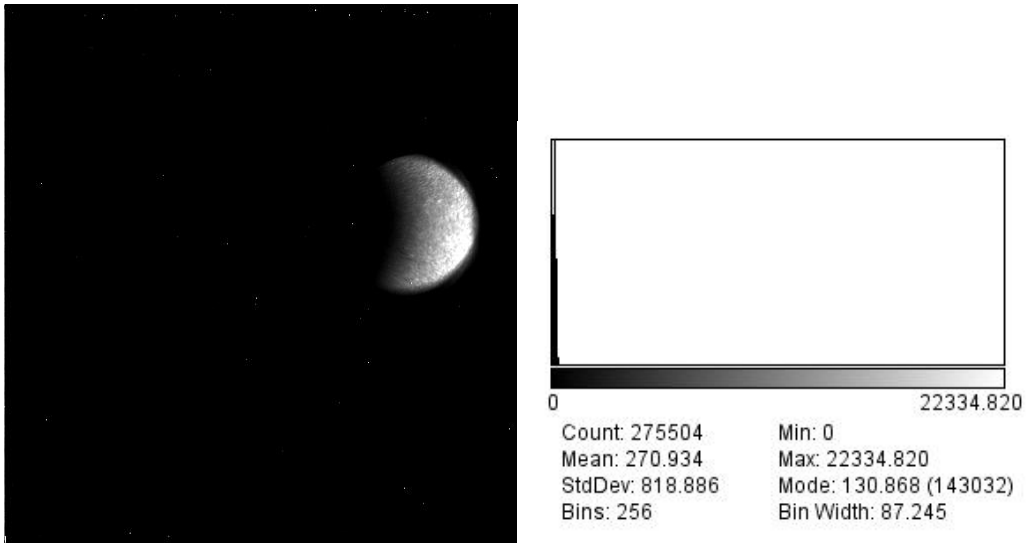

Figure 2.8 – *[Set-up #1] Polarized Brightness*

Selecting a rectangular area (120x150 pixels) including the spot (fig. 2.9), we found:

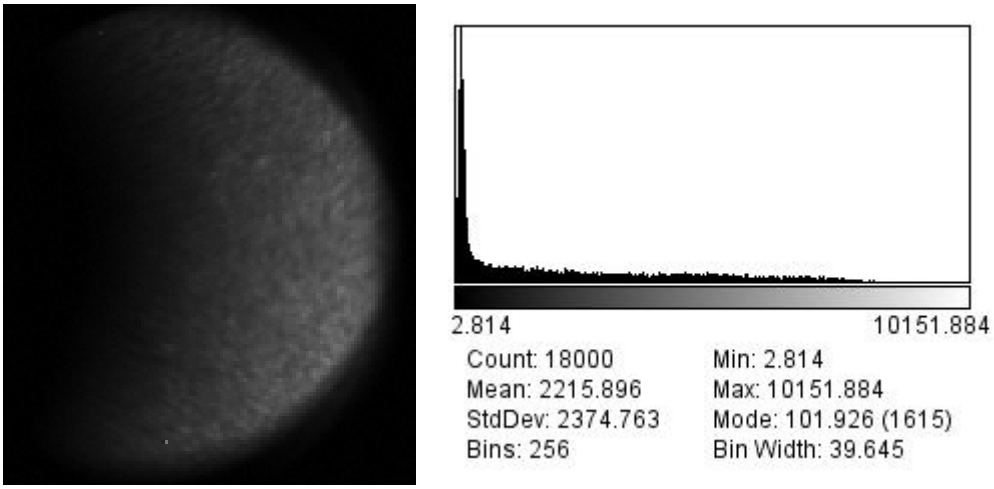

Figure 2.9 – *[Set-up #1] Polarized Brightness Spot*

Profile and surface plot of this spot are in fig. 2.10:

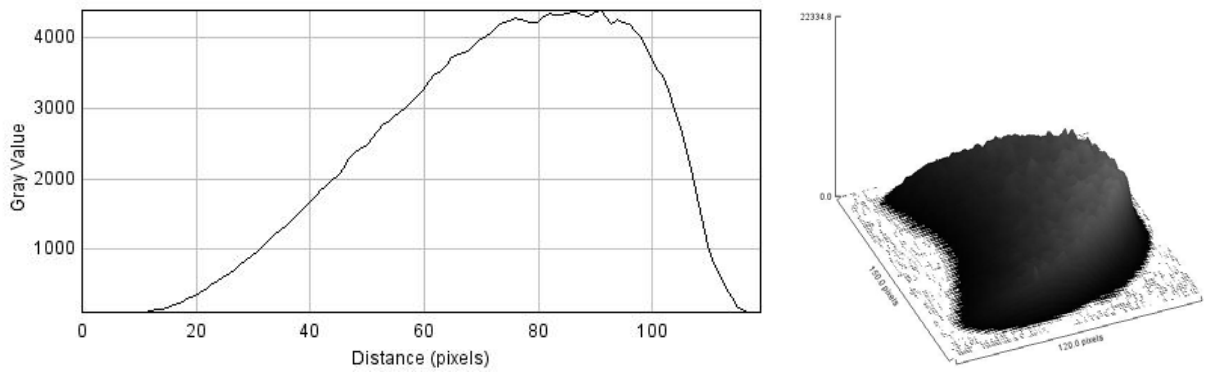

Figure 2.10 – *[Set-up #1] Profile and Surface of the spot*

Resuming, resulting polarized brightness using this set-up is:

pB(1) = 2215 ± 2375 counts (10)

as showed in the histogram of fig. 2.9.

Using data acquired mounting pre-polarizer in front of KPol [set-up #2], we measure:

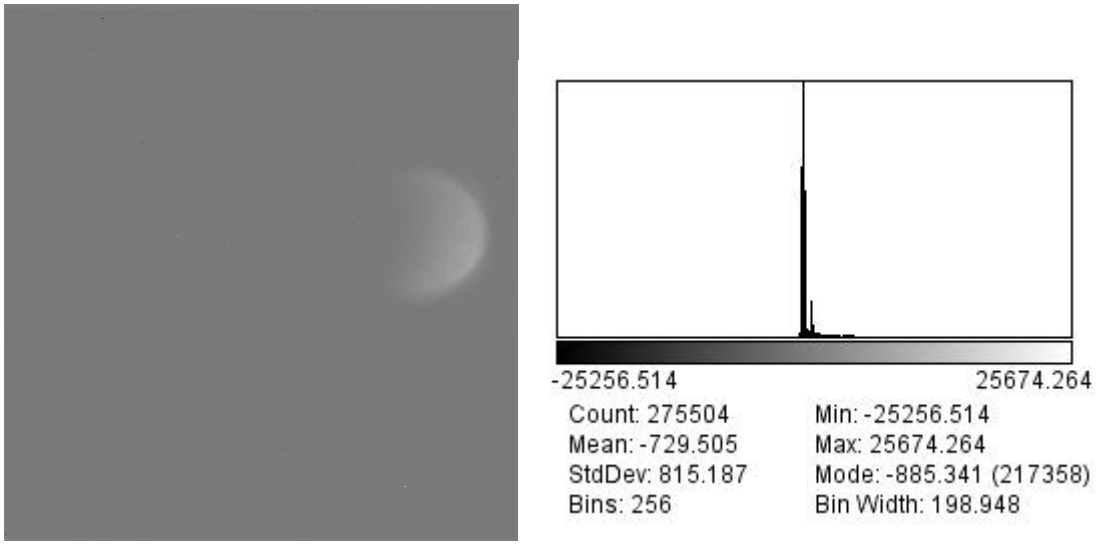

Figure 2.11 – *[Set-up #2] I parameter*

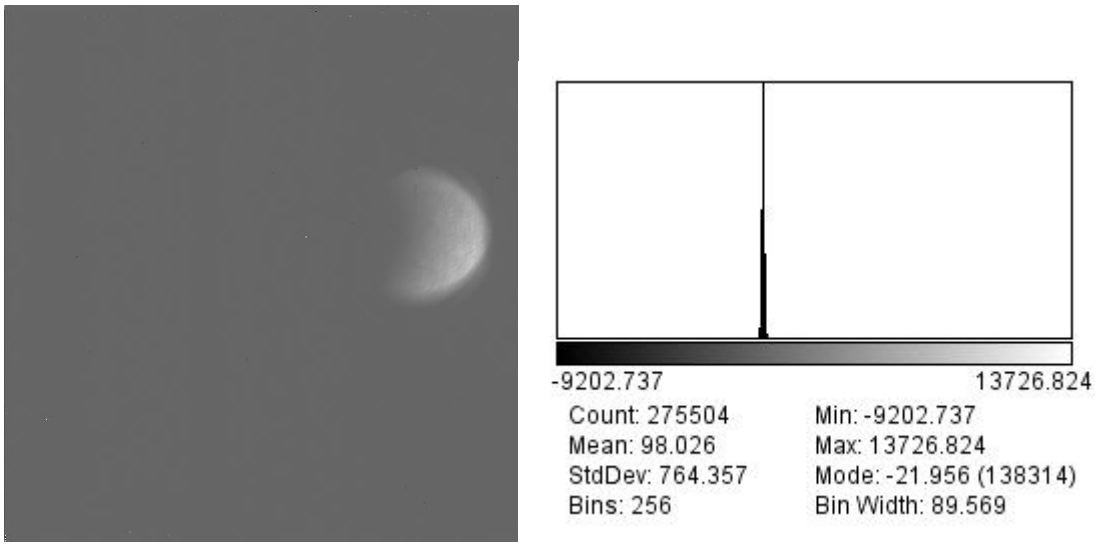

Figure 2.12 – *[Set-up #2] Q parameter*

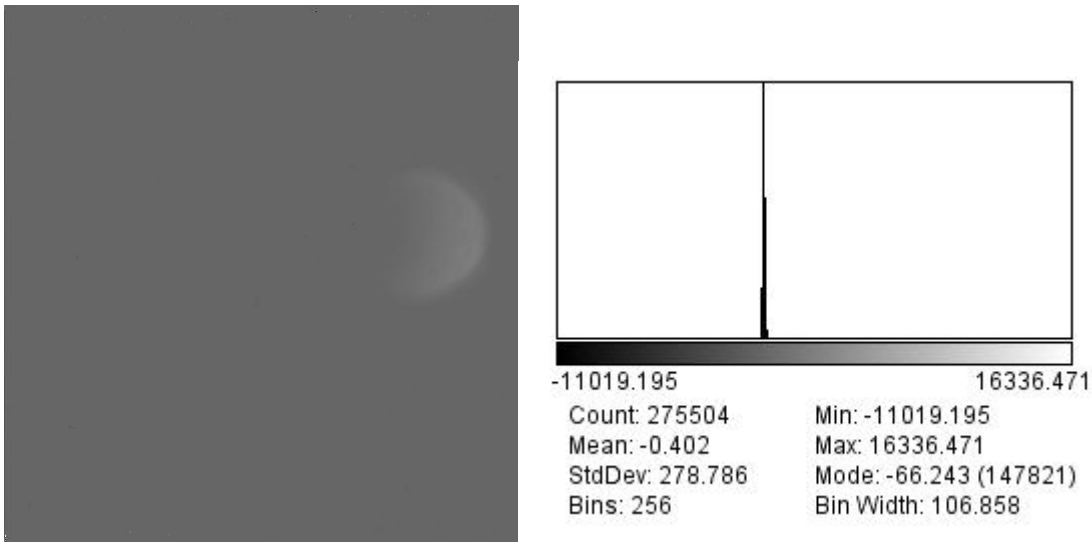

Figure 2.13 – *[Set-up #2] U parameter*

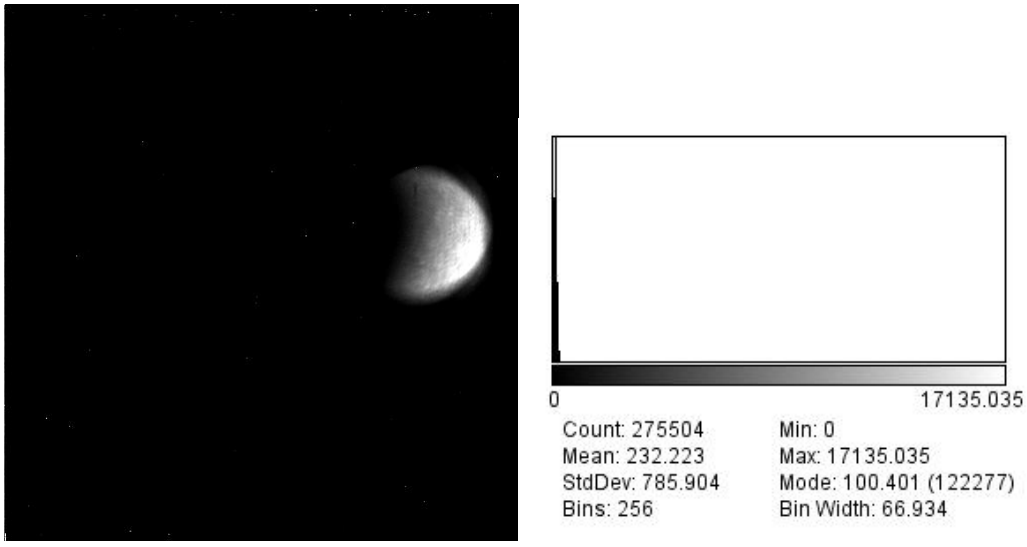

Figure 2.14 – *[Set-up #2] Polarized Brightness*

Selecting a rectangular area (120x150 pixels) including the spot (fig. 2.15), we found:

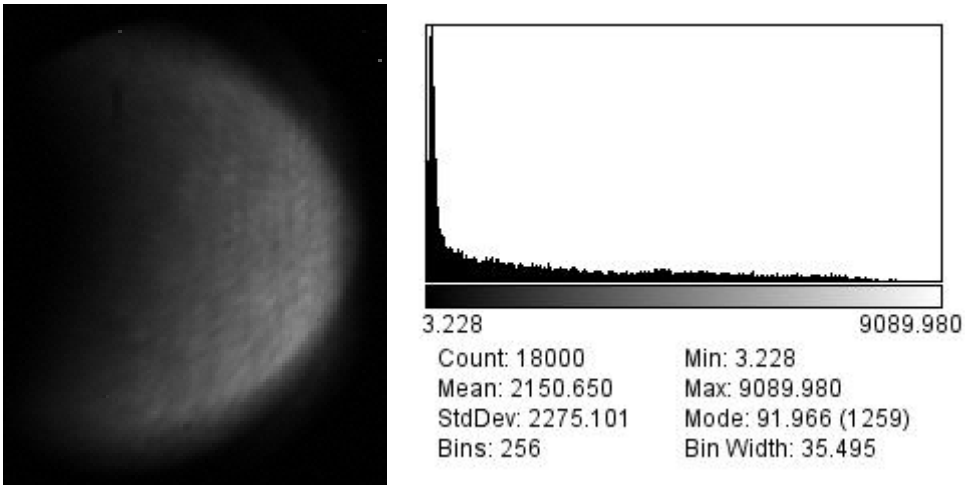

Figure 2.15 – *[Set-up #2] Polarized Brightness Spot*

Profile and surface plot of this spot are in fig. 2.16:

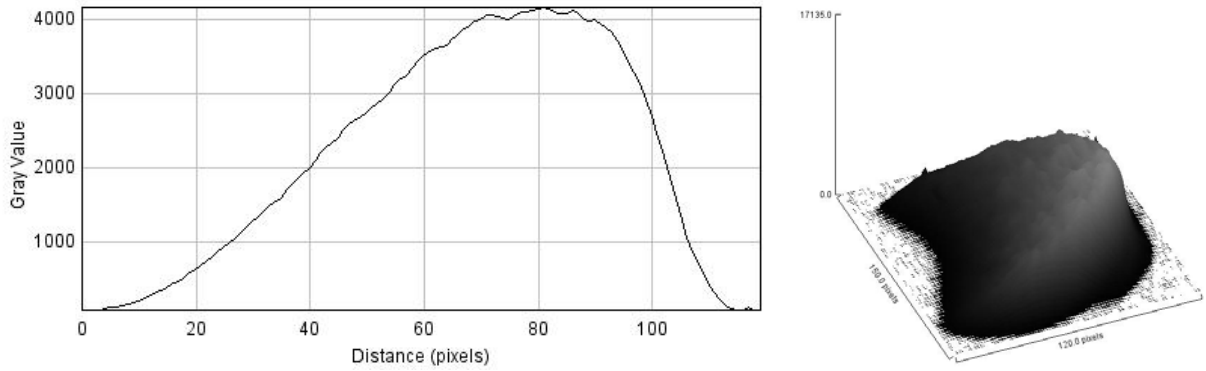

Figure 2.16 – *[Set-up #2] Profile and Surface of the spot*

Resulting polarized brightness using this set-up is:

pB(2) = 2151 ± 2275 counts (11)

as showed in the histogram of fig. 2.15.

Subtracting polarized brightness of set-up #2 to polarized brightness of set-up #1 we obtain instrumental polarization. In formula:

InstrPol = pB(1) – pB(2) = 65 ± 86 counts (12)

Than, instrumental polarization is evaluate equal to 2.9%. Error on this value is compatible with his value, we conclude that instrumental polarization is negligible. Image of resulting instrumental polarization is reported in fig. 2.17.

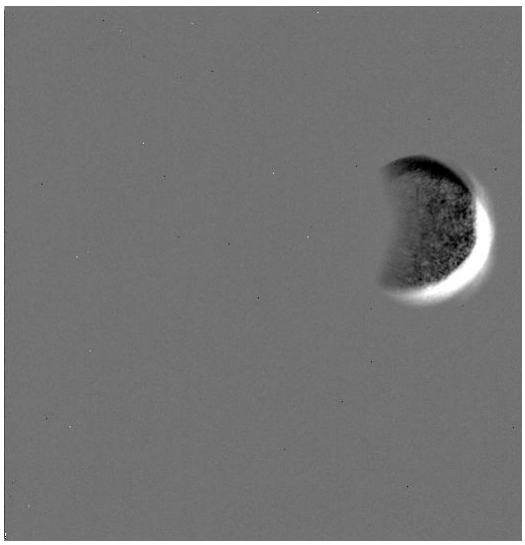

Figure 2.17 – *UVCI Instrumental Polarization.*

Another way for evaluate errors and accuracy of measurements is based over the average counts of the background frames. For background files used for calibration of set-up 2, the mean value is including between 10152 counts (file 88.bkgr_20s_0.fits) and 10067 counts (file 91.bkgr_20s_3.fits). Than error is of 0.8%. Similarly for background frames of set-up 1, the mean values are including between 11234 counts (file 108.bkgr_20s_0.fits) and 11060 counts (file 111.bkgr_20s_3.fits). Error is estimate equal to 1.5%. Error on instrumental polarization is than equal to 1.7%. Histograms of this two set of measurements are reported in fig. 2.18 and 2.19.

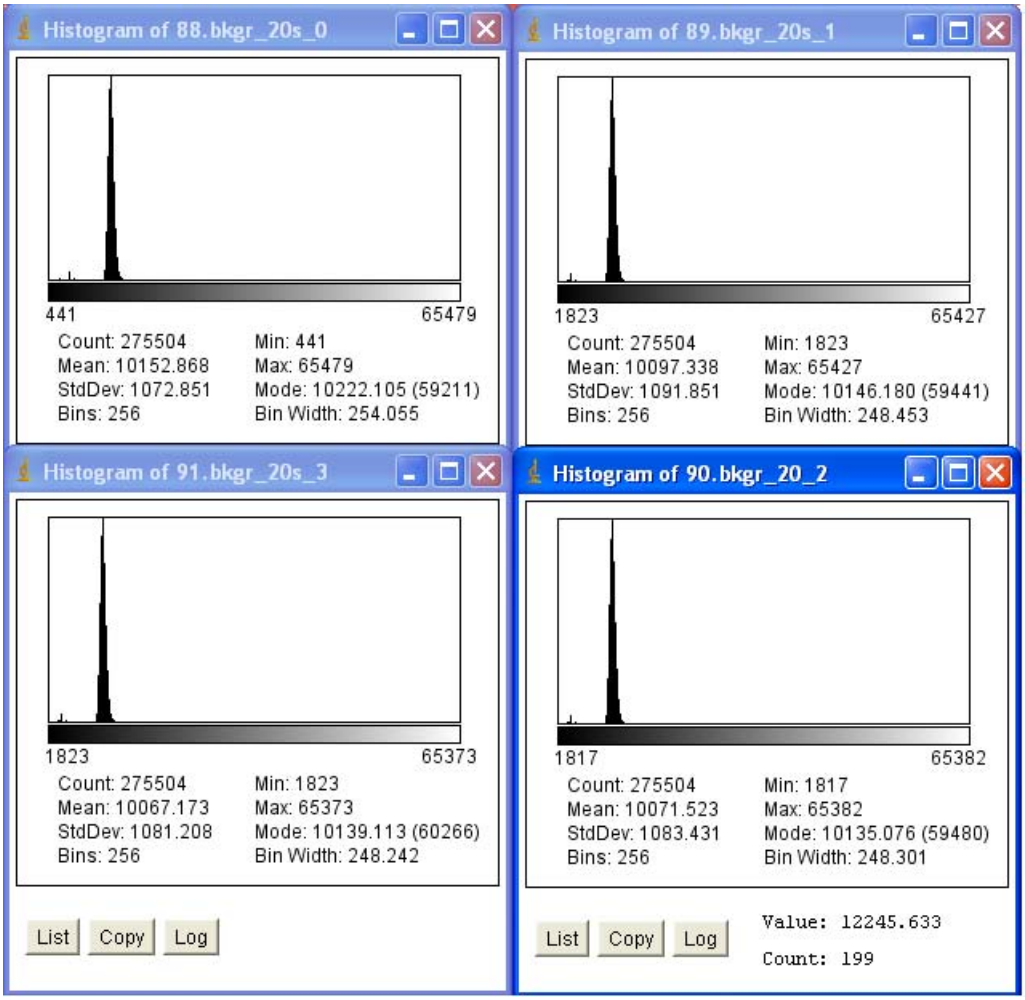

Figure 2.18 – *[Set-up #2] Histograms of background frames.*

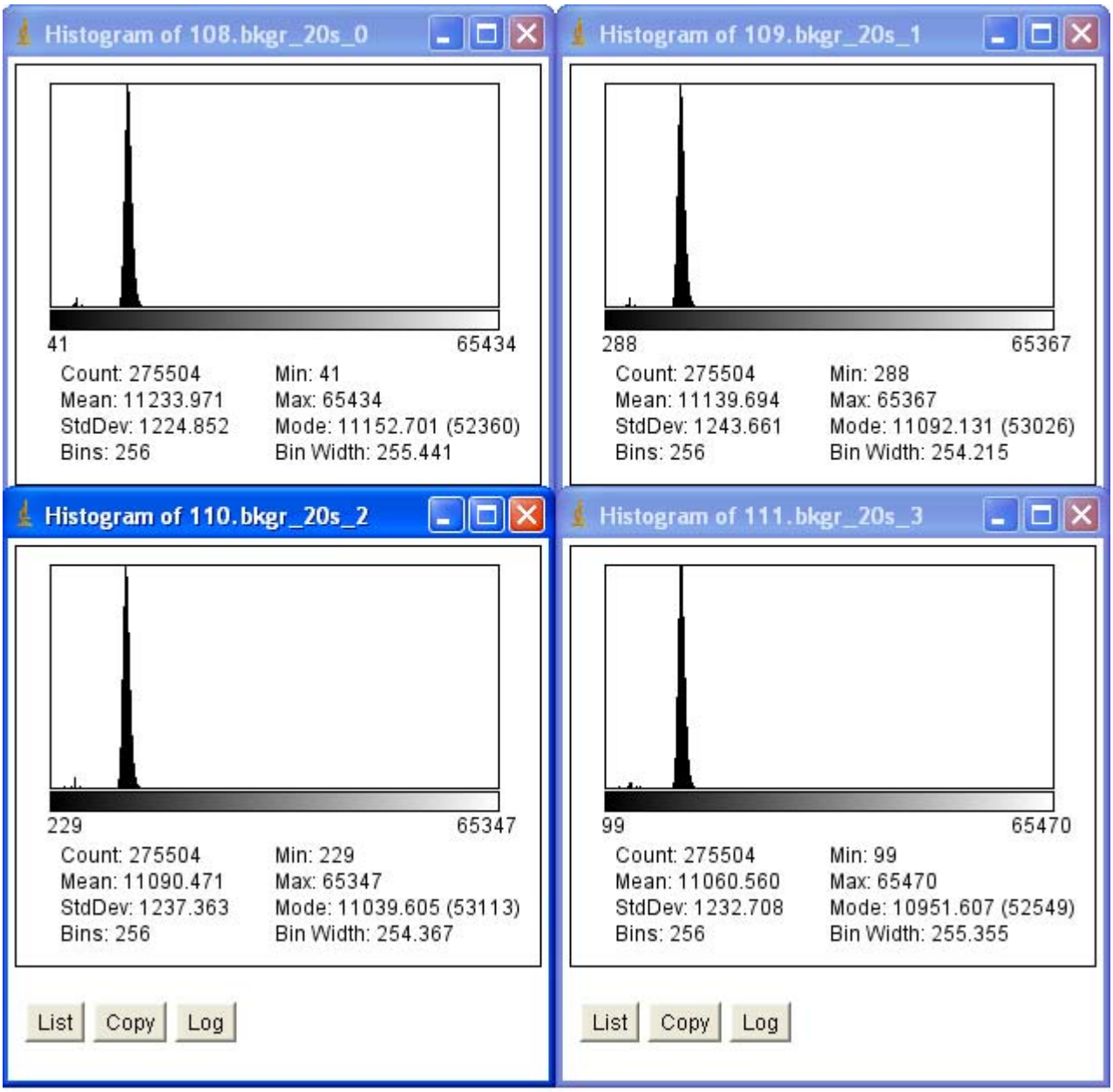

Figure 2.19 – *[Set-up #1] Histograms of background frames.*

ImageJ software has been used for this data analysis [2] . A macro for automatic data analysis is developed. The source code of this macro is in appendix B.

3. Results

Resuming, UVCI instrumental polarizer is evaluate equal to 2.9%. Accuracy of measurement method is of order of 2%. Assuming this as error, instrumental polarization is: 2.9%± 2%.

References

[1] *Capobianco G., Zangrilli L., Fineschi S.*

KPol FM Calibration (LCVR Meadowlark #01-291)

OATo Technical Report 106

[2] ImageJ - Home Page

http://rsb.info.nih.gov/ij/index.html

APPENDIX A – Data File Used for calibration

Files used for evaluation of UVCI instrumental polarization are the follows:

SET-UP #1	Pre-polarizer mounted in front of telescope Filter: 2ND Axis of pre-polarizer at 0 deg = "vertical"

Data frames:

- ../08.08.14_VL_KPol/104.prepol-tel-kpol_20s_ND2_1.fits
- ../08.08.14_VL_KPol/105.prepol-tel-kpol_20s_ND2_2.fits
- ../08.08.14_VL_KPol/106.prepol-tel-kpol_20s_ND2_3.fits
- ../08.08.14_VL_KPol/107.prepol-tel-kpol_20s_ND2_0.fits

Background frames:

- ../08.08.14_VL_KPol/108.bkgr_20s_0.fits
- ../08.08.14_VL_KPol/109.bkgr_20s_1.fits
- ../08.08.14_VL_KPol/110.bkgr_20s_2.fits
- ../08.08.14_VL_KPol/111.bkgr_20s_3.fits

SET-UP #2	Pre-polarizer mounted in front of KPol Filter: 2ND Axis of pre-polarizer at 0 deg = "vertical"

Data frames:

- ../08.08.14_VL_KPol/84.tel-prepol-kpol_20s_ND2_3.fits
- ../08.08.14_VL_KPol/85.tel-prepol-kpol_20s_ND2_0.fits
- ../08.08.14_VL_KPol/86.tel-prepol-kpol_20s_ND2_1.fits
- ../08.08.14_VL_KPol/87.tel-prepol-kpol_20s_ND2_2.fits

Background frames:

- ../08.08.14_VL_KPol/88.bkgr_20s_0.fits
- ../08.08.14_VL_KPol/89.bkgr_20s_1.fits
- ../08.08.14_VL_KPol/90.bkgr_20s_2.fits
- ../08.08.14_VL_KPol/91.bkgr_20s_3.fits

APPENDIX B – ImageJ macro for evaluation of pB

```
////////////////////////////////////////////////////////////////////////////////////////////////
//                                        UVCI_Calib1.2b                                      //
//A imageJ macro for evaluation of pB starting from image @ 0/45/90/135 deg                    //
//                                                                                            //
//   Description: This macro evaluate pB starting from 4 data images and 4 background images  //
//                                                                                            //
//   Author: G. Capobianco                                                                    //
//   Date: 2009-01-11                                                                         //
//   Vers. 1.1                                                                                //
//                                                                                            //
//   Input:   (string) Path data - is directory where input files are located                  //
//            (string) Filename 0 deg - is the filename of 0 deg image                         //
//            (string) Filename bkgr 0 deg - is the filename of 0 deg background image         //
//            (string) Filename 45 deg - is the filename of 45 deg image                       //
//            (string) Filename bkgr 45 deg - is the filename of 45 deg background image       //
//            (string) Filename 90 deg - is the filename of 90 deg image                       //
//            (string) Filename bkgr 90 deg - is the filename of 90 deg background image       //
//            (string) Filename 135 deg - is the filename of 135 deg image                     //
//            (string) Filename bkgr 135 deg - is the filename of 135 deg background image     //
//            (string) Path data out - is directory where save output files                    //
//            (bool)   Remove background - subtract background from data image if True         //
//                                                                                            //
//   Output: (2D - Array)   0deg_calibrated.fits - Fits file of image at 0 deg without background  //
//            (BGR Image) 0deg_calibrated.jpeg - Jpeg of image at 0 deg without background     //
//            (BGR Image) 0deg_hist.jpeg - Histogram of 0 deg calibrated image                //
//            (2D - Array)   45deg_calibrated.fits - Fits file of image at 45 deg without background  //
//            (BGR Image) 45deg_calibrated.jpeg - Jpeg of image at 45 deg without background   //
//            (BGR Image) 45deg_hist.jpeg - Histogram of 45 deg calibrated image              //
//            (2D - Array)   90deg_calibrated.fits - Fits file of image at 90 deg without background  //
//            (BGR Image) 90deg_calibrated.jpeg - Jpeg of image at 90 deg without background   //
//            (BGR Image) 90deg_hist.jpeg - Histogram of 90 deg calibrated image              //
//            (2D - Array)   135deg_calibrated.fits - Fits file of image at 135 deg without background  //
//            (BGR Image) 1355deg_calibrated.jpeg - Jpeg of image at 135 deg without background  //
//            (BGR Image) 135deg_hist.jpeg - Histogram of 135 deg calibrated image            //
//            (2D - Array)   Intensity.fits - Fits file of Intensity (first element of Stockes vector)  //
//            (BGR Image) Intensity.jpeg - Jpeg intensity file                                //
//            (BGR Image) Intensity_hist.jpeg - Histogram of intensity image                  //
//            (2D - Array)   Qpar.fits - Fits file of Q  (2nd element of Stockes vector)      //
//            (BGR Image) Qpar.jpeg - Jpeg Q file                                             //
//            (BGR Image) Qpar_hist.jpeg - Histogram of Q image                               //
//            (2D - Array)   Upar.fits - Fits file of U (3th element of Stockes vector)       //
//            (BGR Image) Upar.jpeg - Jpeg of U                                               //
//            (BGR Image) Upar_hist.jpeg - Histogram of U image                               //
//            (2D - Array)   pB.fits - Fits file of polarized Brightness                      //
//            (BGR Image) pB.jpeg - Jpeg pB file                                              //
//            (BGR Image) pB_hist.jpeg - Histogram of pB image                                //
//                                                                                            //
//           EXAMPLE:                                                                         //
//           Input:C:/Data/Images; image0deg.fits; background.fits; image45deg.fits; background.fits;  //
//                 image90deg.fits; background.fits; image135deg.fits; background.fits;       //
//                 C:/Data/Analysis; True                                                     //
//           Output: C:/Data/Analysis/0deg_calibrated.fits; C:/Data/Analysis/0deg_calibrated.jpg;  //
//                   C:/Data/Analysis/0deg_hist.jpeg; C:/Data/Analysis/45deg_calibrated.fits;  //
//                   C:/Data/Analysis/45deg_calibrated.jpeg;C:/Data/Analysis/45deg_hist.jpg;   //
//                   C:/Data/Analysis/90deg_calibrated.fits; C:/Data/Analysis/90deg_calibrated.jpg;  //
//                   C:/Data/Analysis/90deg_hist.jpeg; C:/Data/Analysis/135deg_calibrated.fits;  //
//                   C:/Data/Analysis/135deg_calibrated.jpeg;C:/Data/Analysis/135deg_hist.jpg;  //
//                   C:/Data/Analysis/Intensity.fits; C:/Data/Analysis/Intensity.jpg;          //
//                   C:/Data/Analysis/Intensity_hist.jpeg; C:/Data/Analysis/Qpar.fits;         //
//                   C:/Data/Analysis/Qpar.jpeg;C:/Data/Analysis/Qpar_hist.jpg;                //
//                   C:/Data/Analysis/Upar.fits; C:/Data/Analysis/Upar.jpg;                    //
```

```
//                      C:/Data/Analysis/Upar_hist.jpeg; C:/Data/Analysis/pB.fits;                    //
//                      C:/Data/Analysis/pB.jpeg;C:/Data/Analysis/pB_hist.jpg;                        //
//                                                                                                    //
//    Comments:                                                                                       //
//    History:       2009/01/14 --> Open also Raw file                                               //
////////////////////////////////////////////////////////////////////////////////////////////////////////
//
 path="C:\Documents and Settings\Gerry\Desktop\UVCI_Instr_pB\08.08.14_VL_KPol";
 filename1 = "85.tel-prepol-kpol_20s_ND2_0.fits";
 filename1b="88.bkgr_20s_0.fits";
 filename2 = "86.tel-prepol-kpol_20s_ND2_1.fits";
 filename2b="89.bkgr_20s_1.fits";
 filename3 = "87.tel-prepol-kpol_20s_ND2_2.fits";
 filename3b="90.bkgr_20_2.fits";
 filename4 = "84.tel-prepol-kpol_20s_ND2_3.fits";
 filename4b="91.bkgr_20s_3.fits";
 pathout="C:\Documents and Settings\Gerry\Desktop\analisi";
//
 Dialog.create("Open Image @ 0/45/90/135 deg ");
 Dialog.addString("Path data:", path);
 Dialog.addString("Filename 0 deg:", filename1);
 Dialog.addString("Filename bkgr 0 deg:", filename1b);
 Dialog.addString("Filename 45 deg:", filename2);
 Dialog.addString("Filename bkgr 45 deg:", filename2b);
 Dialog.addString("Filename 90 deg:", filename3);
 Dialog.addString("Filename bkgr 90 deg:", filename3b);
 Dialog.addString("Filename 135 deg:", filename4);
 Dialog.addString("Filename bkgr 135 deg:", filename4b);
 Dialog.addString("Path data out:", pathout);
 Dialog.addCheckbox("Remove background", true);
 Dialog.show();
 path=Dialog.getString();
 filename1 = Dialog.getString();
 filename1b = Dialog.getString();
 filename2 = Dialog.getString();
 filename2b = Dialog.getString();
 filename3 = Dialog.getString();
 filename3b = Dialog.getString();
 filename4 = Dialog.getString();
 filename4b = Dialog.getString();
 pathout=Dialog.getString();
 rmvbkgr = Dialog.getCheckbox();
 //
 ////////////////////////////////////////////////////
 //return complete filename path //
 ////////////////////////////////////////////////////
 //
 sep=File.separator;
 //
 file1 = path + sep + filename1;
 file1b = path + sep + filename1b;
 file2 = path + sep + filename2;
 file2b = path + sep + filename2b;
 file3 = path + sep + filename3;
 file3b = path + sep + filename3b;
 file4 = path + sep + filename4;
 file4b = path + sep + filename4b;
//
 ////////////////////////////////////////////////
 // Check if files are fits or raw //
 ////////////////////////////////////////////////
 //
  file1fits=endsWith(file1,".fits")&&endsWith(file1,".fit")&&endsWith(file1,".fts");
  file1bfits=endsWith(file1b,".fits")&&endsWith(file1b,".fit")&&endsWith(file1b,".fts");
```

```
  file2fits=endsWith(file2,".fits")&&endsWith(file2,".fit")&&endsWith(file2,".fts");
  file2bfits=endsWith(file2b,".fits")&&endsWith(file2b,".fit")&&endsWith(file2b,".fts");
  file3fits=endsWith(file3,".fits")&&endsWith(file3,".fit")&&endsWith(file3,".fts");
  file3bfits=endsWith(file3b,".fits")&&endsWith(file3b,".fit")&&endsWith(file3b,".fts");
  file4fits=endsWith(file4,".fits")&&endsWith(file4,".fit")&&endsWith(file4,".fts");
  file4bfits=endsWith(file4b,".fits")&&endsWith(file4b,".fit")&&endsWith(file4b,".fts");
 //
 ////////////////////////////////
 //check if files exists //
 ////////////////////////////////
 //
 chk1=File.exists(file1);
 chk1b=File.exists(file1b);
 chk2=File.exists(file2);
 chk2b=File.exists(file2b);
 chk3=File.exists(file3);
 chk3b=File.exists(file3b);
 chk4=File.exists(file4);
 chk4b=File.exists(file4b);
 if(rmvbkgr == 1){
    if (chk1 ==0 || chk1b == 0 || chk2 == 0 || chk2b == 0 || chk3 == 0 || chk3b == 0 || chk4 == 0 || chk4b == 0)
     {exit("one or more file(s) not found");}}
    else{
    if (chk1 ==0 || chk2 == 0 || chk3 == 0 || chk4 == 0)
     {exit("one or more file(s) not found");}}
 //
 ////////////////////////////////////////////////////////////////////////////
 //open selected files, remove bkgr, save calibrated and histogram files and close files  //
 ////////////////////////////////////////////////////////////////////////////
 //
 //Calibrate image @ 0 deg
 //
 if(file1fits==1)
  {open(file1);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file1+imp_para;
  run("Raw...", prop);}
 id1=getImageID();
 img1= getTitle;
//
 if(rmvbkgr == 1){
 if(file1bfits==1)
  {open(file1b);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file1b+imp_para;
  run("Raw...", prop);}
 id1b=getImageID();
 img1b= getTitle;
 //
 imageCalculator("subtract create", img1, img1b);
 id1c=getImageID();}
 else{id1c=id1;}
 sl1=nSlices;
 rename("Image @ 0 deg - Calibrated");
 file0degcalpath=pathout+sep+"0deg_calibrated.fits";
 saveAs ("fits",pathout+sep+"0deg_calibrated.fits");
 saveAs ("jpeg",pathout+sep+"0deg_calibrated.jpeg");
 //
 //Create histogram
 //
 if (bitDepth==8 || bitDepth==24) {
    min = 0;
```

```
    max = 255;
} else {
    if (bitDepth==16)
        getStatistics(nPixels, mean, min, max);
   else
        getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
    ymax = parseFloat(ymax);
bins = 256;
n = nSlices;
setBatchMode(true);
if (n==1) {
     setKeyDown("alt");
     run("Histogram", "bins="+bins+" x_min="+xmin
        +" x_max="+xmax+" y_max="+ymax);
} else {
    stack = getImageID;
    histograms = 0;
    for (i=1; i<=n; i++) {
        showProgress(i, n);
        selectImage(stack);
        setSlice(i);
        setKeyDown("alt");
        run("Histogram", "bins="+bins+" x_min="+xmin
            +" x_max="+xmax+" y_max="+ymax);
        run("Copy");
        width=getWidth; height=getHeight;
        type = "8-bit"; if (bitDepth==24) type = "RGB";
        close();
        if (histograms==0) {
            newImage("Histogram", type, width, height, n);
            histograms = getImageID;
        }
        selectImage(histograms);
        setSlice(i);
        run("Paste");
    }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"0deg_hist.jpeg");
id1h=getImageID();
//  print("0deg: mean= "+mean+"  max= "+max+"  min= "+min);
//FIne istogramma
if(rmvbkgr == 1){
selectImage(id1);
close();
selectImage(id1b);
close();}
selectImage(id1h);
close();
//
//Calibrate image @ 45 deg
//
if(file2fits==1)
  {open(file2);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file2+imp_para;
  run("Raw...", prop);}
```

```
 id2=getImageID();
 img2= getTitle;
//
 if(rmvbkgr == 1){
 if(file2bfits==1)
  {open(file2b);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file2b+imp_para;
  run("Raw...", prop);}
 id2b=getImageID();
 img2b= getTitle;
 //
 imageCalculator("subtract create", img2, img2b);
 id2c=getImageID();}else{id2c=id2;}
 rename("Image @ 45 deg - Calibrated");
 file45degcalpath=pathout+sep+"45deg_calibrated.fits";
 saveAs ("fits",pathout+sep+"45deg_calibrated.fits");
 saveAs ("jpeg",pathout+sep+"45deg_calibrated.jpeg");
 //
 //Create histogram
 //
 if (bitDepth==8 || bitDepth==24) {
    min = 0;
    max = 255;
 } else {
    if (bitDepth==16)
       getStatistics(nPixels, mean, min, max);
   else
       getRawStatistics(nPixels, mean, min, max);
 }
 xmin = min;
 xmax = max;
 ymax="auto";
 if (ymax!="auto")
    ymax = parseFloat(ymax);
 bins = 256;
 n = nSlices;
 setBatchMode(true);
 if (n==1) {
     setKeyDown("alt");
     run("Histogram", "bins="+bins+" x_min="+xmin
       +" x_max="+xmax+" y_max="+ymax);
 } else {
    stack = getImageID;
    histograms = 0;
    for (i=1; i<=n; i++) {
       showProgress(i, n);
       selectImage(stack);
       setSlice(i);
       setKeyDown("alt");
       run("Histogram", "bins="+bins+" x_min="+xmin
          +" x_max="+xmax+" y_max="+ymax);
       run("Copy");
       width=getWidth; height=getHeight;
       type = "8-bit"; if (bitDepth==24) type = "RGB";
       close();
       if (histograms==0) {
          newImage("Histogram", type, width, height, n);
          histograms = getImageID;
       }
       selectImage(histograms);
       setSlice(i);
       run("Paste");
```

```
   }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"45deg_hist.jpeg");
id2h=getImageID();
//  print("45deg mean= "+mean+"  max= "+max+"  min= "+min);
//FIne istogramma
if(rmvbkgr == 1){
selectImage(id2);
close();
selectImage(id2b);
close();}
selectImage(id2h);
close();
//
//Calibrate image @ 90 deg
//
if(file3fits==1)
  {open(file3);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file3+imp_para;
  run("Raw...", prop);}
id3=getImageID();
img3= getTitle;
if(rmvbkgr == 1){
if(file3bfits==1)
  {open(file3b);}
  else{
  imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
  prop="open="+file3b+imp_para;
  run("Raw...", prop);}
id3b=getImageID();
img3b= getTitle;
//
imageCalculator("subtract create", img3, img3b);
id3c=getImageID();}else{id3c=id3;}
rename("Image @ 90 deg - Calibrated");
file90degcalpath=pathout+sep+"90deg_calibrated.fits";
saveAs ("fits",pathout+sep+"90deg_calibrated.fits");
saveAs ("jpeg",pathout+sep+"90deg_calibrated.jpeg");
//
//Create histogram
//
if (bitDepth==8 || bitDepth==24) {
    min = 0;
    max = 255;
} else {
    if (bitDepth==16)
        getStatistics(nPixels, mean, min, max);
   else
        getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
    ymax = parseFloat(ymax);
bins = 256;
n = nSlices;
setBatchMode(true);
if (n==1) {
     setKeyDown("alt");
```

```
    run("Histogram", "bins="+bins+" x_min="+xmin
      +" x_max="+xmax+" y_max="+ymax);
} else {
   stack = getImageID;
   histograms = 0;
   for (i=1; i<=n; i++) {
      showProgress(i, n);
      selectImage(stack);
      setSlice(i);
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
         +" x_max="+xmax+" y_max="+ymax);
      run("Copy");
      width=getWidth; height=getHeight;
      type = "8-bit"; if (bitDepth==24) type = "RGB";
      close();
      if (histograms==0) {
         newImage("Histogram", type, width, height, n);
         histograms = getImageID;
      }
      selectImage(histograms);
      setSlice(i);
      run("Paste");
   }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"90deg_hist.jpeg");
id3h=getImageID();
//FIne istogramma
if(rmvbkgr == 1){
selectImage(id3);
close();
selectImage(id3b);
close();}
selectImage(id3h);
close();
//
//Calibrate image @ 135 deg
//
if(file4fits==1)
 {open(file4);}
 else{
 imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
 prop="open="+file4+imp_para;
 run("Raw...", prop);}
id4=getImageID();
img4= getTitle;
if(rmvbkgr == 1){
if(file4bfits==1)
 {open(file4b);}
 else{
 imp_para=" image=[16-bit Unsigned] width=536 height=514 offset=7 number=1 gap=0  white is zero ";
 prop="open="+file4b+imp_para;
 run("Raw...", prop);}
id4b=getImageID();
img4b= getTitle;
//
imageCalculator("subtract create", img4, img4b);
id4c=getImageID();}else{id4c=id4;}
rename("Image @ 135 deg - Calibrated");
file135degcalpath=pathout+sep+"135deg_calibrated.fits";
saveAs ("fits",pathout+sep+"135deg_calibrated.fits");
saveAs ("jpeg",pathout+sep+"135deg_calibrated.jpeg");
```

```
//
//Create histogram
//
if (bitDepth==8 || bitDepth==24) {
   min = 0;
   max = 255;
} else {
   if (bitDepth==16)
      getStatistics(nPixels, mean, min, max);
  else
      getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
   ymax = parseFloat(ymax);
bins = 256;
n = nSlices;
setBatchMode(true);
if (n==1) {
    setKeyDown("alt");
    run("Histogram", "bins="+bins+" x_min="+xmin
      +" x_max="+xmax+" y_max="+ymax);
} else {
   stack = getImageID;
   histograms = 0;
   for (i=1; i<=n; i++) {
      showProgress(i, n);
      selectImage(stack);
      setSlice(i);
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
         +" x_max="+xmax+" y_max="+ymax);
      run("Copy");
      width=getWidth; height=getHeight;
      type = "8-bit"; if (bitDepth==24) type = "RGB";
      close();
      if (histograms==0) {
         newImage("Histogram", type, width, height, n);
         histograms = getImageID;
      }
      selectImage(histograms);
      setSlice(i);
      run("Paste");
   }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"135deg_hist.jpeg");
id4h=getImageID();
//  print("135deg mean= "+mean+"  max= "+max+"  min= "+min);
//FIne istogramma
if(rmvbkgr == 1){
selectImage(id4);
close();
selectImage(id4b);
close();}
selectImage(id4h);
close();
//
////////////////////////////
//Evaluate Intensity image //
////////////////////////////
```

```
//
// Values of a^(-1) = p
// first row
p11 = 0.267627;
p12 = 0.246785;
p13 = 0.233534;
p14 = 0.252055;
// second row
p21 = 0.516075;
p22 = -0.0235569;
p23 = -0.480393;
p24 = -0.0121246;
// third row
p31 = 0.0636487;
p32 = 0.495361;
p33 = -0.0472306;
p34 = -0.511779;
//
// return m0 * p11
//
selectImage(id1c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p11);
  }
updateDisplay();
rename("m0*p11");
img01=getTitle;
//
// return m1 * p12
//
selectImage(id2c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p12);
  }
updateDisplay();
rename("m1*p12");
img11=getTitle;
//
// return m2 * p13
//
selectImage(id3c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p13);
  }
updateDisplay();
rename("m2*p13");
img21=getTitle;
//
// return m3 * p14
//
selectImage(id4c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
```

```
    setPixel(x, y, getPixel(x, y)*p14);
   }
  updateDisplay();
  rename("m3*p14");
  img31=getTitle;
//
  imageCalculator("add create", img01, img11);
  updateDisplay();
  rename("1st sum");
  imgI0=getTitle;
  idI0=getImageID();
  imageCalculator("add create", img21, img31);
  updateDisplay();
  rename("2nd sum");
  imgI1=getTitle;
  idI1=getImageID();
  imageCalculator("add create", imgI0,imgI1);
  updateDisplay();
  //
  rename("Intensity");
  imgInt=getTitle;
  idInt=getImageID();
  //
  saveAs ("fits",pathout+sep+"Intensity.fits");
  saveAs ("jpeg",pathout+sep+"Intensity.jpeg");
  //
  //Create histogram
  //
  if (bitDepth==8 || bitDepth==24) {
     min = 0;
     max = 255;
  } else {
     if (bitDepth==16)
        getStatistics(nPixels, mean, min, max);
    else
        getRawStatistics(nPixels, mean, min, max);
  }
  xmin = min;
  xmax = max;
  ymax="auto";
  if (ymax!="auto")
     ymax = parseFloat(ymax);
  bins = 256;
  n = nSlices;
  setBatchMode(true);
  if (n==1) {
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
        +" x_max="+xmax+" y_max="+ymax);
  } else {
     stack = getImageID;
     histograms = 0;
     for (i=1; i<=n; i++) {
        showProgress(i, n);
        selectImage(stack);
        setSlice(i);
        setKeyDown("alt");
        run("Histogram", "bins="+bins+" x_min="+xmin
           +" x_max="+xmax+" y_max="+ymax);
        run("Copy");
        width=getWidth; height=getHeight;
        type = "8-bit"; if (bitDepth==24) type = "RGB";
        close();
        if (histograms==0) {
```

```
            newImage("Histogram", type, width, height, n);
            histograms = getImageID;
        }
        selectImage(histograms);
        setSlice(i);
        run("Paste");
    }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"Intensity_hist.jpeg");
close();
//FIne istogramma
selectImage(idI0);
close();
selectImage(idI1);
close();
selectImage(id1c);
close();
selectImage(id2c);
close();
selectImage(id3c);
close();
selectImage(id4c);
close();
//
// Rapro le immagini
//
open(pathout+sep+"0deg_calibrated.fits");
id1c=getImageID();
open(pathout+sep+"45deg_calibrated.fits");
id2c=getImageID();
open(pathout+sep+"90deg_calibrated.fits");
id3c=getImageID();
open(pathout+sep+"135deg_calibrated.fits");
id4c=getImageID();
//////////////////////
//Evaluate Q image //
//////////////////////
//
//
// return m0 * p21
//
selectImage(id1c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p21);
  }
updateDisplay();
rename("m0*p21");
img02=getTitle;
//
// return m1 * p22
//
selectImage(id2c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p22);
  }
updateDisplay();
```

```
rename("m1*p22");
img12=getTitle;
//
// return m2 * p23
//
selectImage(id3c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p23);
  }
updateDisplay();
rename("m2*p23");
img22=getTitle;
//
// return m3 * p24
//
selectImage(id4c);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, getPixel(x, y)*p24);
  }
updateDisplay();
rename("m3*p24");
img32=getTitle;
//
imageCalculator("add create", img02, img12);
updateDisplay();
rename("1st sum");
imgQ0=getTitle;
idQ0=getImageID();
imageCalculator("add create", img22, img32);
updateDisplay();
rename("2nd sum");
imgQ1=getTitle;
idQ1=getImageID();
imageCalculator("add create", imgQ0,imgQ1);
updateDisplay();
//
rename("Q parameter");
imgQpar=getTitle;
idQ= getImageID();
//
saveAs ("fits",pathout+sep+"Qpar.fits");
saveAs ("jpeg",pathout+sep+"Qpar.jpeg");
//
//Create histogram
//
if (bitDepth==8 || bitDepth==24) {
    min = 0;
    max = 255;
} else {
    if (bitDepth==16)
        getStatistics(nPixels, mean, min, max);
   else
        getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
```

```
    ymax = parseFloat(ymax);
  bins = 256;
  n = nSlices;
  setBatchMode(true);
  if (n==1) {
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
        +" x_max="+xmax+" y_max="+ymax);
  } else {
     stack = getImageID;
     histograms = 0;
     for (i=1; i<=n; i++) {
        showProgress(i, n);
        selectImage(stack);
        setSlice(i);
        setKeyDown("alt");
        run("Histogram", "bins="+bins+" x_min="+xmin
           +" x_max="+xmax+" y_max="+ymax);
        run("Copy");
        width=getWidth; height=getHeight;
        type = "8-bit"; if (bitDepth==24) type = "RGB";
        close();
        if (histograms==0) {
           newImage("Histogram", type, width, height, n);
           histograms = getImageID;
        }
        selectImage(histograms);
        setSlice(i);
        run("Paste");
     }
  }
  run("Select None");
  setBatchMode(false);
  saveAs ("jpeg",pathout+sep+"Qpar_hist.jpeg");
  close();
  //  idIh=getImageID();
  //FIne istogramma
  selectImage(idQ0);
  close();
  selectImage(idQ1);
  close();
  selectImage(id1c);
  close();
  selectImage(id2c);
  close();
  selectImage(id3c);
  close();
  selectImage(id4c);
  close();
  //
  // Rapro le immagini
  //
  open(pathout+sep+"0deg_calibrated.fits");
  id1c=getImageID();
  open(pathout+sep+"45deg_calibrated.fits");
  id2c=getImageID();
  open(pathout+sep+"90deg_calibrated.fits");
  id3c=getImageID();
  open(pathout+sep+"135deg_calibrated.fits");
  id4c=getImageID();
//
  ////////////////////
  //Evaluate U image //
  ////////////////////
```

```
  //
  // return m0 * p31
  //
  selectImage(id1c);
  w = getWidth();
  h = getHeight();
  for (y=0; y<h; y++) {
    for (x=0; x<w; x++)
      setPixel(x, y, getPixel(x, y)*p31);
    }
  updateDisplay();
  rename("m0*p31");
  img03=getTitle;
  //
  // return m1 * p32
  //
  selectImage(id2c);
  w = getWidth();
  h = getHeight();
  for (y=0; y<h; y++) {
    for (x=0; x<w; x++)
      setPixel(x, y, getPixel(x, y)*p32);
    }
  updateDisplay();
  rename("m1*p32");
  img13=getTitle;
  //
  // return m2 * p33
  //
  selectImage(id3c);
  w = getWidth();
  h = getHeight();
  for (y=0; y<h; y++) {
    for (x=0; x<w; x++)
      setPixel(x, y, getPixel(x, y)*p33);
    }
  updateDisplay();
  rename("m2*p33");
  img23=getTitle;
  //
  // return m3 * p34
  //
  selectImage(id4c);
  w = getWidth();
  h = getHeight();
  for (y=0; y<h; y++) {
    for (x=0; x<w; x++)
      setPixel(x, y, getPixel(x, y)*p34);
    }
  updateDisplay();
  rename("m3*p34");
  img33=getTitle;
//
  imageCalculator("add create", img03, img13);
  updateDisplay();
  rename("1st sum");
  imgU0=getTitle;
  idU0=getImageID();
  imageCalculator("add create", img23, img33);
  updateDisplay();
  rename("2nd sum");
  imgU1=getTitle;
  idU1=getImageID();
  imageCalculator("add create", imgU0,imgU1);
```

```
updateDisplay();
//
rename("U parameter");
imgUpar=getTitle;
idU= getImageID();
//
saveAs ("fits",pathout+sep+"Upar.fits");
saveAs ("jpeg",pathout+sep+"Upar.jpeg");
//
//Create histogram
//
if (bitDepth==8 || bitDepth==24) {
   min = 0;
   max = 255;
} else {
   if (bitDepth==16)
      getStatistics(nPixels, mean, min, max);
  else
      getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
   ymax = parseFloat(ymax);
bins = 256;
n = nSlices;
setBatchMode(true);
if (n==1) {
    setKeyDown("alt");
    run("Histogram", "bins="+bins+" x_min="+xmin
      +" x_max="+xmax+" y_max="+ymax);
} else {
   stack = getImageID;
   histograms = 0;
   for (i=1; i<=n; i++) {
      showProgress(i, n);
      selectImage(stack);
      setSlice(i);
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
         +" x_max="+xmax+" y_max="+ymax);
      run("Copy");
      width=getWidth; height=getHeight;
      type = "8-bit"; if (bitDepth==24) type = "RGB";
      close();
      if (histograms==0) {
         newImage("Histogram", type, width, height, n);
         histograms = getImageID;
      }
      selectImage(histograms);
      setSlice(i);
      run("Paste");
   }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"Upar_hist.jpeg");
close();
//  idIh=getImageID();
//FIne istogramma
selectImage(idU0);
close();
selectImage(idU1);
```

```
close();
selectImage(id1c);
close();
selectImage(id2c);
close();
selectImage(id3c);
close();
selectImage(id4c);
close();
 //
// Rapro le immagini
//
open(pathout+sep+"0deg_calibrated.fits");
id1c=getImageID();
open(pathout+sep+"45deg_calibrated.fits");
id2c=getImageID();
open(pathout+sep+"90deg_calibrated.fits");
id3c=getImageID();
open(pathout+sep+"135deg_calibrated.fits");
id4c=getImageID();
//
//////////////////////
//Evaluate pB image //
//////////////////////
//
selectImage(idQ);
imageCalculator("multiply create", imgQpar, imgQpar);
updateDisplay();
rename("Q square");
imgQsq=getTitle;
idQsq=getImageID();
selectImage(idU);
imageCalculator("multiply create", imgUpar, imgUpar);
updateDisplay();
rename("U square");
imgUsq=getTitle;
idUsq=getImageID();
imageCalculator("add create", imgUsq,imgQsq);
updateDisplay();
idpBsq=getImageID();
selectImage(idpBsq);
w = getWidth();
h = getHeight();
for (y=0; y<h; y++) {
  for (x=0; x<w; x++)
    setPixel(x, y, sqrt(getPixel(x, y)));
  }
updateDisplay();
rename("polarizedBrightness");
imgpB=getTitle;
idpB= getImageID();
//
saveAs ("fits",pathout+sep+"pB.fits");
saveAs ("jpeg",pathout+sep+"pB.jpeg");
//
//Create histogram
//
if (bitDepth==8 || bitDepth==24) {
    min = 0;
    max = 255;
} else {
    if (bitDepth==16)
        getStatistics(nPixels, mean, min, max);
   else
```

```
      getRawStatistics(nPixels, mean, min, max);
}
xmin = min;
xmax = max;
ymax="auto";
if (ymax!="auto")
   ymax = parseFloat(ymax);
bins = 256;
n = nSlices;
setBatchMode(true);
if (n==1) {
    setKeyDown("alt");
    run("Histogram", "bins="+bins+" x_min="+xmin
      +" x_max="+xmax+" y_max="+ymax);
} else {
   stack = getImageID;
   histograms = 0;
   for (i=1; i<=n; i++) {
      showProgress(i, n);
      selectImage(stack);
      setSlice(i);
      setKeyDown("alt");
      run("Histogram", "bins="+bins+" x_min="+xmin
         +" x_max="+xmax+" y_max="+ymax);
      run("Copy");
      width=getWidth; height=getHeight;
      type = "8-bit"; if (bitDepth==24) type = "RGB";
      close();
      if (histograms==0) {
         newImage("Histogram", type, width, height, n);
         histograms = getImageID;
      }
      selectImage(histograms);
      setSlice(i);
      run("Paste");
   }
}
run("Select None");
setBatchMode(false);
saveAs ("jpeg",pathout+sep+"pB_hist.jpeg");
close();
selectImage(idQsq);
close();
selectImage(idUsq);
close();
//
////////////////////////////////////////
// Show images and close it after time ms //
////////////////////////////////////////
//
time= 3000;
selectImage(id1c);
wait(time);
close();
selectImage(id2c);
wait(time);
close();
selectImage(id3c);
wait(time);
close();
selectImage(id4c);
wait(time);
close();
selectImage(idInt);
```

```
wait(time);
close();
selectImage(idQ);
wait(time);
close();
selectImage(idU);
wait(time);
close();
selectImage(idpB);
wait(time);
close();
//
//END
//
beep();
exit("Data Analysis is complete. Data are saved in "+ pathout);
```