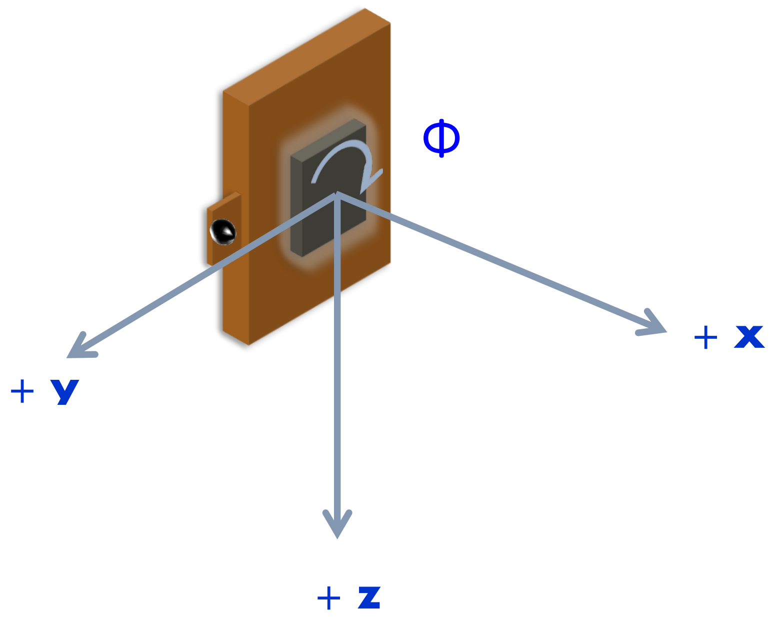
I21 Diffcalc Guide



Crystal orientation

Before moving in hkl space you must calculate a UB matrix by specifying the crystal's lattice parameters (which define the B matrix) and finding two reflections (from which the U matrix can be inferred); and, optionally for surface-diffraction experiments, determine how the surface of the crystal is oriented with respect to the phi axis.

Start a new UB calculation

A *UB calculation* contains the description of the crystal-under-test, any saved reflections, reference angle direction, and a B & UB matrix pair if they have been calculated or manually specified. Starting a new UB calculation will clear all of these.

Before starting a UB-calculation, the `ub` command used to summarise the state of the current UB-calculation, will reflect that no UB-calculation has been started:

```
>>> ub
<<< No UB calculation started >>>
```

A new UB-calculation calculation may be started and lattice specified explicitly:

```
>>> newub 'example'
>>> setlat '1Acube' 1 1 1 90 90 90
```

or interactively:

```
>>> newub
calculation name: example
crystal name: 1Acube
crystal system
1) Triclinic
2) Monoclinic
3) Orthorhombic
4) Tetragonal
5) Rhombohedral
6) Hexagonal
7) Cubic
[1]: 7
    a[1]: 1
```

where a is unit cell basis vector in Angstroms for cubic crystal system.

The `ub` command will show the state of the current UB-calculation (and the current energy for reference):

```
>>> ub
UBCALC

   name:       example

REFERNCE

   n_hkl:      1.00000   0.00000   0.00000 <- set

SURFACE NORMAL

   n_phi:      0.00000   0.00000   1.00000 <- set

CRYSTAL

   name:        1Acube

   a, b, c:    1.00000   1.00000   1.00000
              90.00000  90.00000  90.00000  Cubic

   B matrix:   6.28319   0.00000   0.00000
               0.00000   6.28319   0.00000
               0.00000   0.00000   6.28319

UB MATRIX

   <<< none calculated >>>

REFLECTIONS

   <<< none specified >>>

CRYSTAL ORIENTATIONS

   <<< none specified >>>
```

Load a UB calculation

To load the last used UB-calculation:

```
>>> lastub
Loading ub calculation: 'mono-Si'
```

To load a previous UB-calculation:

```
>>> listub
UB calculations in: /home/i21user/.diffcalc/i21

0) mono-Si            15 Feb 2017 (22:32)
1) i21-32             13 Feb 2017 (18:32)

>>> loadub 0
```

Generate a U matrix from two lattice directions

Another approach to calculate a U matrix is to provide orientation of **two** crystal lattice directions using `addorient` command after aligning sample in laboratory frame of reference. The first lattice direction should be aligned along the selected direction in the laboratory frame. Best would be to provide the lattice direction normal to the surface of the sample. For the purpose of finding azimuthal orientation in U matrix calculation it is sufficient for the projection of the second lattice direction to be aligned to the given orientation in the laboratory frame in the plane perpendicular to the first lattice orientation.

Find U matrix from two lattice directions:

```
>>> addorient [0 0 1] [1 0 0]

>>> addorient [1 0 0] [0 1 0]
Calculating UB matrix.
```

Calculate a UB matrix

Unless a U or UB matrix has been manually specified, a new UB matrix will be calculated after the second reflection has been found, or whenever one of the first two reflections is changed.

Use command `orientub` to force the UB matrix to be calculated from the first two orientations.

UB matrix can be calculated from any combination of two reflections and/or orientations by providing corresponding reflection/orientation tags or numbers as an argument to `calcub`. In case of using one reflection and one orientation it is recommended to use tags to avoid ambiguity.

If you have misidentified a reflection used for the orientation the resulting UB matrix will be incorrect. Always use the `checkub` command to check that the computed reflection indices agree with the estimated values:

```
>>> checkub

     ENERGY     H     K     L     H_COMP   K_COMP   L_COMP      TAG
 1  12.3984  0.00  1.00  1.00    0.0000   1.0000   1.0000
 2  12.3984  0.00  0.00  1.00    0.0000   0.0000   1.0000
```

Calculate a U matrix from crystal mismount

U matrix can be defined from crystal mismount by using a rotation matrix calculated from a provided mismount angle and axis. `setmiscut` command defines new U matrix by setting it to a rotation matrix calculated from the specified angle and axis parameters. `addmiscut` command applies the calculated rotation matrix to the existing U matrix, i.e. adds extra mismount to the already existing one:

```
>>> setmiscut 5 [1 0 0]
n_phi: -0.00000  -0.08716   0.99619
n_hkl:  0.00000   0.00000   1.00000 <- set
normal:
   angle:  5.00000
   axis:  1.00000  -0.00000   0.00000
```

Manually specify U matrix

Set U matrix manually (pretending sample is squarely mounted):

```
>>> setu [[1 0 0] [0 1 0] [0 0 1]]
Recalculating UB matrix.
NOTE: A new UB matrix will not be automatically calculated when the
orientation reflections are modified.
```

Set the reference vector

The reference vector can be used to define azimuthal direction within the crystal with which we want to orient the incident or diffracted beam. Orientation of the reference vector w.r.t the incident and diffracted beam is indicated using `alpha` and `beta` angles.

By default in i21 the reference vector is set parallel to the theta axis. That is, along the y-axis of the laboratory coordinate frame.

The `ub` command shows the current reference vector at the top its report (or it can be shown by calling `setnphi` or `setnhkl` with no args):

```
>>> ub
...
   REFERNCE
   n_phi:       0.00000   1.00000   0.00000
   n_hkl:       1.00000   0.00000   0.00000 <- set
...
```

The `<- set` label here indicates that the reference vector is set in the reciprocal lattice space. In this case, therefore, its direction in the laboratory coordinate frame is inferred from the UB matrix.

To set the reference vector in the phi coordinate frame use:

```
>>> setnphi [0 1 0]
...
```

To set the reference vector in the crystal's reciprocal lattice space use:

```
>>> setnhkl [0 1 0]
...
```

Constraining solutions for moving in hkl space

To get help and see current constraints:

```
>>> help con
...

>>> con
      REF             SAMP
    -----------     -----------
    a_eq_b          th
    alpha           chi
    beta            phi
    psi             mu_is_gam
    bin_eq_bout     bisect
    betain          omega
    betaout

!   1 more constraint required

    Type 'help con' for instructions
```

REFERENCE COLUMN:

- **alpha** - incident angle to reference vector
- **beta** - exit angle from reference vector
- **psi** - azimuthal rotation about scattering vector of reference vector (from scattering plane)
- **a_eq_b** - bisecting mode with alpha=beta. *Equivalent to psi=90*
- **betain** - incident angle to sample surface
- **betaout** - exit angle from sample surface

SAMPLE COLUMN:

- **mu, eta, chi & phi** - physical settings
- **mu_is_gam** - force mu to follow gamma (results in a 5-circle geometry)
- **bisect** - bisecting mode with scattering vector in chi-circle plane
- **omega** - bisecting mode with omega angle between scattering vector and chi-circle plane

Diffcalc will report two other (un-constrainable) virtual angles:

- **theta** - half of 2theta, the angle through the diffracted beam bends
- **tau** - longitude of reference vector from scattering vector (in scattering plane)

Configuring limits and cuts

Diffcalc uses motor limits set in GDA when used from GDA client running on a beamline. The standalone console version maintains its own limits on axes. These limits will be used when choosing solutions. If more than one detector solution exists Diffcalc will ask you to reduce the the limits until there is only one. However if more than one solution for the sample settings is available it will choose one that is closest to the current diffractometer orientation.

Use the `hardware` command to see the current limits and cuts:

```
>>> hardware
            mu          (cut: -180.0)
         delta          (cut: -180.0)
           gam          (cut: -180.0)
           eta          (cut: -180.0)
           chi          (cut: -180.0)
           phi          (cut:    0.0)
Note: When auto sector/transforms are used,
       cuts are applied before checking limits.
```

To set the limits in standalone Diffcalc session:

```
>>> setmin delta -1
>>> setmax delta 145
```

To set a cut:

```
>>> setcut phi -180
```

This causes requests to move phi to be between the configured -180 and +360 degress above this. i.e. it might dive to -10 degrees rather than 350.

Configuring reciprocal space as function of selected detector

Due to the unique requirements of I21 RIXS spectrometer we have the possibility of using different detectors for diffcalc. For the aligning of the sample crystal axes using diffraction we can use the rotating photodiode that is located inside the sample vessel and whose motor is called `difftth`. Once the aligment is completed and we move to the spectrometer to perform RIXS measurements, then we have the possibility to refer our geometry either to the collecting mirror `m5tth` or to any of the two collecting mirrors that we call `lowq` and `highq`.
To be able to select between this available geometries, we have a set of scannables `hkl_m5tth`, `hkl_lowq`, `hkl_highq` and `hkl_difftth` that relate to different combinations of selected angles and geometries. They can be used as `hkl` scannable to do calculations and move diffractometer. The `hkl` can be also reassigned to one of these scannables for convenience using `usem5tth`, `uselowq`, `usehighq` and `usedifftth` commands.

Moving in hkl space

Configure a mode, e.g. four-circle vertical:

```
>>> con psi 0
    psi  : 0.0000
```

Simulate moving to a reflection:

```
>>>  sim hkl [0 0 0.25]
_fourc would move to:
    delta :   76.3430
       th :   38.1715
      chi :    0.0000
      phi :    0.0000

    alpha :  -51.8285
     beta :   51.8285
   betain :   38.1715
  betaout :   38.1715
      naz :    0.0000
      psi :    0.0000
      qaz :    0.0000
      tau :   90.0000
    theta :   38.1715
   ttheta :   76.3430
```

Move to reflection:

```
>>> pos hkl [0 0 0.25]
hkl:      h: 0.00000 k: 0.00000 l: 0.25000
```

As we explained before since in i21 we have four different available geometries, it is important to make sure we are choosing the right one before moving to the reflection as the positions would be different for the different geometries.

```
>>>  usedifftth
- setting hkl ---> hkl_difftth
Loading ub calculation: 'manual'
```

```
WARNING: Ignoring constraint eta
WARNING: Ignoring constraint delta
INFO: diffcalc limits set in $diffcalc/startup/i21.py taken
from http://confluence.diamond.ac.uk/pages/viewpage.action?pageId=51413586
Current hardware limits set to:
     0.0 <= delta <=  180.0 (cut: -180.0)
     0.0 <=    th <=  150.0 (cut:    0.0)
   -41.0 <=   chi <=   36.0 (cut: -180.0)
  -100.0 <=   phi <=  100.0 (cut: -180.0)
Note: When auto sector/transforms are used,
       cuts are applied before checking limits.


>>>  sim hkl [0 0 0.25]
_fourc would move to:
    delta :   76.3430
       th :   38.1715
      chi :    0.0000
      phi :    0.0000

    alpha :  -51.8285
     beta :   51.8285
   betain :   38.1715
  betaout :   38.1715
      naz :    0.0000
      psi :    0.0000
      qaz :    0.0000
      tau :   90.0000
    theta :   38.1715
   ttheta :   76.3430
```

But if we choose a different geometrical configuration:

```
>>>  usehighq
- setting hkl ---> hkl_highq
Loading ub calculation: 'manual'
WARNING: Ignoring constraint eta
WARNING: Ignoring constraint delta
INFO: diffcalc limits set in $diffcalc/startup/i21.py taken
from http://confluence.diamond.ac.uk/pages/viewpage.action?pageId=51413586
Current hardware limits set to:
     0.0 <= delta <=  180.0 (cut: -180.0)
     0.0 <=    th <=  150.0 (cut:    0.0)
   -41.0 <=   chi <=   36.0 (cut: -180.0)
  -100.0 <=   phi <=  100.0 (cut: -180.0)
Note: When auto sector/transforms are used,
       cuts are applied before checking limits.


>>>  sim hkl [0 0 0.25]
_fourc would move to:
    delta :   72.3430
       th :   38.1715
      chi :    0.0000
      phi :    0.0000

    alpha :  -51.8285
```

```
      beta :   51.8285
    betain :   38.1715
   betaout :   38.1715
       naz :    0.0000
       psi :    0.0000
       qaz :    0.0000
       tau :   90.0000
     theta :   38.1715
    ttheta :   76.3430
```

Note that the delta positions are different depending which detector we use as geometrical reference.

Scanning energy keeping the Q constant. EfixQ

If we want to measure a energy dependence but keeping always the same geometry of a Q as a function of energy, first we need to align on the Q that we want to keep constant.

```
>>> energy
  energy:  643.0000

>>> pos hkl [0 0 0.25]
hkl_diffth:h: -0.00000 k: -0.00000 l: 0.25000

>>> hkl

hkl:
  hkl_diffth :   -0.0000  -0.0000  0.2500

     alpha :  -51.8285
      beta :   51.8285
    betain :   38.1715
   betaout :   38.1715
       naz :    0.0000
       psi :    0.0000
       qaz :    0.0000
       tau :   90.0000
     theta :   38.1715
    ttheta :   76.3430

_fourc:
     delta :   76.3430
        th :   38.1715
       chi :    0.0000
       phi :    0.0000
```

Then to scan the energy

```
>>> scan energy 636 660 1 ct 0.1 hkl [0 0 0.25] fourc
       en         h          k         l     delta         th      chi      phi       ct
---------  -------  --------  -------  --------  --------  -------  -------  -------
 636.0000  0.00000  -0.00000  0.25000   77.3379   38.6689   0.0000   0.0000  0.32485
 637.0000  0.00000  -0.00000  0.25000   77.1940   38.5970   0.0000   0.0000  0.32485
 638.0000  -0.00000  -0.00000  0.25000   77.0507   38.5254   0.0000   0.0000  0.32485
 639.0000  -0.00000  -0.00000  0.25000   76.9080   38.4540   0.0000   0.0000  0.32485
 640.0000  0.00000  -0.00000  0.25000   76.7659   38.3829   0.0000   0.0000  0.32485
```

```
641.0000  -0.00000  -0.00000  0.25000   76.6244   38.3122   0.0000   0.0000  0.32485
642.0000  -0.00000  -0.00000  0.25000   76.4834   38.2417   0.0000   0.0000  0.32485
643.0000  -0.00000  -0.00000  0.25000   76.3430   38.1715   0.0000   0.0000  0.32485
644.0000  -0.00000  -0.00000  0.25000   76.2032   38.1016   0.0000   0.0000  0.32485
645.0000  -0.00000  -0.00000  0.25000   76.0640   38.0320   0.0000   0.0000  0.32485
646.0000  -0.00000  -0.00000  0.25000   75.9253   37.9626   0.0000   0.0000  0.32485
647.0000  0.00000  -0.00000  0.25000   75.7871   37.8936   0.0000   0.0000  0.32485
648.0000  -0.00000  -0.00000  0.25000   75.6496   37.8248   0.0000   0.0000  0.32485
649.0000  -0.00000  -0.00000  0.25000   75.5126   37.7563   0.0000   0.0000  0.32485
650.0000  -0.00000  -0.00000  0.25000   75.3761   37.6880   0.0000   0.0000  0.32485
651.0000  0.00000  -0.00000  0.25000   75.2402   37.6201   0.0000   0.0000  0.32485
652.0000  -0.00000  -0.00000  0.25000   75.1048   37.5524   0.0000   0.0000  0.32485
653.0000  -0.00000  -0.00000  0.25000   74.9699   37.4850   0.0000   0.0000  0.32485
654.0000  -0.00000  -0.00000  0.25000   74.8356   37.4178   0.0000   0.0000  0.32485
655.0000  0.00000  -0.00000  0.25000   74.7018   37.3509   0.0000   0.0000  0.32485
656.0000  -0.00000  -0.00000  0.25000   74.5686   37.2843   0.0000   0.0000  0.32485
657.0000  0.00000  -0.00000  0.25000   74.4358   37.2179   0.0000   0.0000  0.32485
658.0000  0.00000  -0.00000  0.25000   74.3036   37.1518   0.0000   0.0000  0.32485
659.0000  0.00000  -0.00000  0.25000   74.1719   37.0860   0.0000   0.0000  0.32485
660.0000  -0.00000  -0.00000  0.25000   74.0407   37.0204   0.0000   0.0000  0.32485
```

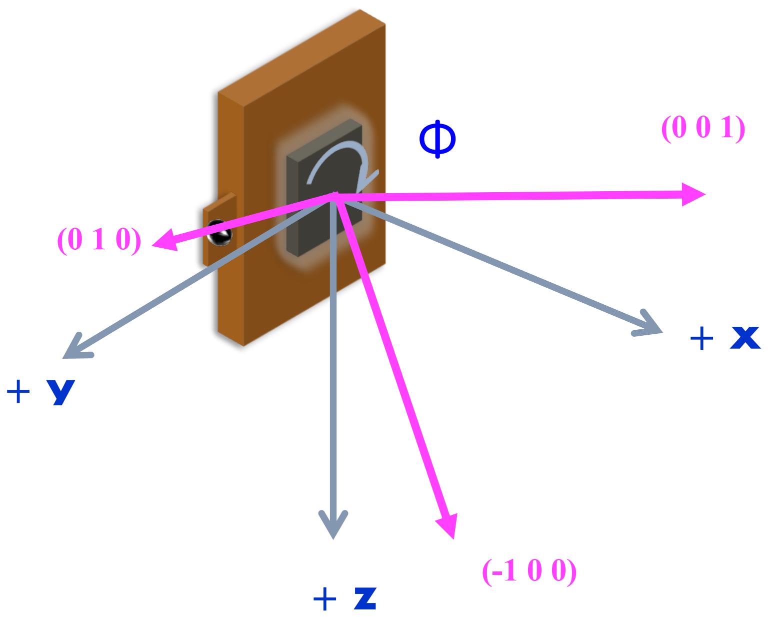
Where `hkl [0 0 0.25]` is the **Q** that we want to keep fixed, `ct` is the counter that needs to be substituted by the names of the counters we want to use in each case. By including `fourc` at the end of the scan, GDA will print on the command line the values of `delta`, `th`, `chi` and `phi` for each point of the scan.

Moving to different azimuthal angles keeping the Q constant.

Often when we align a particular Q direction for one sample we find that there are certain offsets in `th`, `chi` or `phi` that need to be taken into account in order to be properly aligned.
If we want to change geometry from one azimuthal angle to another keeping the specular geometry fixed, diffcalc can help us doing this while taking the offsets into account. For instance if we want to move from (pi,0) geometry to (pi,pi) geometry.
Remember that

- **psi** - azimuthal rotation about scattering vector of reference vector (from scattering plane)

Often when we align the scattering plane we know the offsets in `phi` only by the `laue` measurements as most of the times we cannot access any in plane reflection to refine it using diffraction.
Imaging a case as the one below when the scattering plane is forming an angle with the laboratory axes.



```
>>> newub
calculation name: uboffset
crystal name: SBMO
crystal system
1) Triclinic
2) Monoclinic
3) Orthorhombic
4) Tetragonal
5) Rhombohedral
6) Hexagonal
7) Cubic
[1]: 7
    a[1]: 3.9
```

```
>>> addorient
h[0.0]: 0
k[0.0]: 0
l[0.0]: 1
x[0.0]: 1
y[0.0]: 0
z[0.0]: 0
current pos[y]: n
  delta[74.0407334683]: 0
     th[37.3675946909]: 0
    chi[10.0]: 10
    phi[5.0]: 5
tag: [001]

>>> In [121]: addorient
h[0.0]: 0
k[0.0]: 1
l[0.0]: 0
x[0.0]: 0
y[0.0]: 1
z[0.0]: 0
current pos[y]: n
  delta[74.0407334683]: 0
     th[37.3675946909]: 0
    chi[10.0]: 10
    phi[5.0]: 5
tag: [010]
Calculating UB matrix.
```

Here we have included the offsets we know from the `Laue` measurements: `phi = 5 deg.` and `chi = 10 deg.` while defining the orientation reflections that will create our `ub matrix`.

```
>>> ub
UBCALC

   name:      uboffset

REFERNCE

   n_phi:      0.17365  -0.08583  -0.98106
   n_hkl:      1.00000   0.00000   0.00000 <- set

SURFACE NORMAL

   n_phi:      1.00000   0.00000   0.00000 <- set
   n_hkl:      0.17365   0.00000   0.98481

CRYSTAL

   name:          SBMO

   a, b, c:    3.90000   3.90000   3.90000
              90.00000  90.00000  90.00000  Cubic

   B matrix:   0.00000   0.00000   1.61107
               0.00000   1.61107   0.00000
              -1.61107   0.00000   0.00000

UB MATRIX
```

```
   U matrix:    0.98481   0.00000  -0.17365
                0.01513   0.99619   0.08583
                0.17299  -0.08716   0.98106

   miscut:
      angle:  10.00000
       axis:    0.00000  -0.99619   0.08716

   UB matrix:  0.27976   0.00000   1.58660
              -0.13828   1.60494   0.02438
              -1.58056  -0.14041   0.27870

REFLECTIONS

   <<< none specified >>>

CRYSTAL ORIENTATIONS

        H     K     L       X     Y     Z      DELTA       TH       CHI       PHI  TAG
   1  0.00  0.00  1.00   1.00  0.00  0.00    0.0000   0.0000  10.0000   5.0000  [001]
   2  0.00  1.00  0.00   0.00  1.00  0.00    0.0000   0.0000  10.0000   5.0000  [010]
```

We want to define the `(pi,0)` position when the plane of scattering coincides with the plane fcontained by the `[001]` and `[010]` directions. In this case since there is an offset between this plane and the `phi=0` position, we need to consider as a reference for the azimuthal rotation the `[010]` crystal axes and not the laboratory reference. For doing this:

```
>>> setnhkl [0 1 0]
   n_phi: -0.00000   0.99619  -0.08716
   n_hkl:  0.00000   1.00000   0.00000 <- set
```

Then we constrain the azimuthal angle so that the (pi,0) plane coincides with the plane of scattering. Based on how we have defined the ub matrix, in this case it would be psi = 0.

```
>>> con psi 0
    psi  : 0.0000

>>> pos en 643
en:        643.0000

>>> sim hkl [0 0 0.25]
_fourc would move to:
    delta :   76.3430
       th :   38.1715
      chi :   10.0000
      phi :    5.0000

    alpha :  -51.8285
     beta :   51.8285
   betain :   37.4904
  betaout :   37.4904
      naz :   -0.0000
      psi :    0.0000
      qaz :    0.0000
      tau :   90.0000
    theta :   38.1715
   ttheta :   76.3430
```

```
>>> pos hkl [0 0 0.25]
hkl_difftth:h: -0.00000 k: -0.00000 l: 0.25000
```

This will move the motors to the `(pi,0)` plane taking into account all offsets.

If now we want to move to the `(pi,pi)` scattering plane, then we just need to constrain the azimuthal angle for `45 deg.`

```
>>> con psi 45
    psi  : 45.0000

>>> sim hkl [0 0 0.25]
_fourc would move to:
    delta :   76.3430
       th :   31.0644
      chi :    7.0530
      phi :   50.4385

    alpha :  -33.7729
     beta :   33.7729
   betain :   30.8036
  betaout :   44.8425
      naz :  -58.2832
      psi :   45.0000
      qaz :    0.0000
      tau :   90.0000
    theta :   38.1715
   ttheta :   76.3430

>>> pos hkl [0 0 0.25]
hkl_difftth:h: -0.00000 k: -0.00000 l: 0.25000
```

This will move the motors to the `(pi,pi)` plane taking into account all offsets.

Analogously we can use this system to move to any other athimuthal we want considering all the offsets.