ALTERNATIVE ITTERATIVE ALIGNMENT METHOD FOR PC-DMIS BY NEIL CHALLINOR (HEXAGON MI)

A common requirement is for parts to be aligned from datum target points / features. If the pattern of the points satisfy the rules set out in the PC-Dmis help files, then the iterative routine within the standard alignment dialogue can be used. This routine essentially works on a plane – line – point basis whereby the first three points / features supplied form the planar element of the alignment and will be used for the Level, the next two points / features form the linear element of the alignment and are used for rotation and the last point / feature acts as the origin. The vectors for the first three points / features must be roughly parallel to each other, the vectors of the next two points / features must be roughly perpendicular to the vectors for the first three points and the vector for the last point / feature must be roughly perpendicular to both previous sets of points.

Obviously these rules mean that it is not always possible to use the standard iterative alignment routine – consider for example a part that requires the level to be along the axis of a cone / line or cylinder. Since the routine is expecting a minimum of three points with roughly parallel vectors it will not work.

In scenarios such as this, it is necessary to adopt a different approach.

Iterative alignment routines work by repeatedly measuring a series of points / features, refining the accuracy of the alignment with each iteration until certain criteria are met and the alignment is deemed to be acceptable. With the built in routine these criteria are -

1) The fixture tolerance which limits the amount of deviation for the points / features used for the level and only comes into play if more than 3 points / features are used – think of this as the flatness tolerance for your plane.

2) The point target tolerance. This is how far from nominal the points / features used for the alignment are allowed to drift.

We can adopt a similar approach by aligning the part and then dimensioning the features / points used to see how far from nominal they are and then incorporating some logic into the program to jump back and re-measure if they are not close enough. This could involve quite a substantial chunk of code depending on the number of points / features you have used since you would need to test every axis of ever feature. A simpler method is to create generic “test” points at the extreme limits of your part, measure the features and create the alignment, then dimension the “test” points to see how far from nominal they are and employ logic to re-measure if necessary. The next few pages will detail the steps necessary to create such an alignment on a part that uses an axis for the level – a typical example would be a turbine blade for a jet engine.

Step 1 – Import the Cad Model

Although it is not strictly necessary to work from a CAD model, you must ensure that all nominal values are correct an exact to at least 3 decimal places or you will be introducing error into your alignment and it may fail to resolve or give incorrect results. For the purpose of this example we will assume you are using a Cad model. Import the model in the usual way and note the position and orientation of the CAD co-ordinate system - this is the alignment you are trying to create.

Step 2 – Save the Alignment

It may seem strange to save the alignment before you have created it but don't forget that your program is going to loop round and repeat the alignment process a number of times. This initial alignment save is merely a starting point.

Step 3 – Begin your Alignment Sequence

Since your program is going to loop round a number of times it needs to know where to begin its sequence so you need to insert a label (INSERT>FLOW CONTROL>LABLE). A label is simply a marker that can be referenced programmatically.

Next, recall the alignment you saved earlier. Once we have finished creating the alignment we will save the alignment with the same name, overwriting the original. By recalling the alignment at the beginning of each iteration we are able to refine it. This saving / recalling as also necessary to "trick" PC-Dmis into not updating the nominal values of your program. Since it doesn't see the alignment change because it is saved externally under the same name each time PC-Dmis will not try to correct for this.

Now create two generic points – one at each end of your part.

Step 4 – Measure the Features

Proceed to measure all of the features required to build your alignment. It may be necessary to include multiple sub-alignments along the way if the part is very complex or you need to control the location of one feature in relation to another but the beauty of this method is that this is all possible and you are only constrained by what is possible within PC-Dmis.

Once you have measured all of your features, build your alignment being sure to constrain all six degrees of freedom via the correct use of LEVEL, ROTATE and ORIGIN.

The finished alignment must EXACTLY match the original co-ordinate system of the CAD model that was saved at the beginning of the program. It may be necessary to include additional translations and rotations after your level, rotate and origin steps.

STEP 5 – Save the Alignment Again

Now that we have finished the alignment we need to save it again under the same name as before. As stated earlier, this repeated updating of the alignment is how we refine the accuracy with each iteration.

STEP 6 – Test for Acceptance

Now we need to dimension the two "test" points created at the beginning of our sequence. Use a simple location dimension and report X, Y & Z.

Once you have dimensioned the “test” points you can insert the logic that will test for acceptance. This is done by using an IF_GOTO command (INSERT>FLOW CONTROL>IF_GOTO). This command is basically two commands in one. The IF part performs a logical decision based on the expression that is entered. The GOTO part is linked to the label you created earlier. If the expression equates to TRUE then the program execution jumps back to the label. You will need six IF_GOTO commands in total – one for each axis of each dimension for the “test” points. The syntax would look something like this –

IF_GOTO/ABS(TEST_P1_OP50.X.DEV)>0.003,GOTO = OP1_START_AL

Where “ABS(TEST_P1_OP50.X.DEV)>0.003 “ is the expression to test and “OP1_START_AL” is the name of the label to jump back to. The 0.003 value is the amount (± since it is an absolute value) of deviation that the test points are allowed to have from there nominal positions. If the amount of deviation exceeds this value, then we jump back to our label and re-measure / re-align our part.

Full Syntax Example

```
        SAVE/ALIGNMENT,JJ115234_S_R_R.aln,MACHINETOPARTS
OP1_START_ALN=LABEL/
        RECALL/ALIGNMENT,EXTERNAL,JJ115234_S_R_R
OP1_MAIN_ALIGN=ALIGNMENT/START,RECALL:JJ115234_S_R_R,LIST=YES
        ALIGNMENT/END
OP1_TEST1  =FEAT/POINT,CARTESIAN,NO
        THEO/<0,0,0>,<0,0,1>
        ACTL/<0,0,0>,<0,0,1>
        CONSTR/POINT,ORIGIN
OP1_TEST2  =GENERIC/POINT,DEPENDENT,CARTESIAN,$
        NOM/XYZ,<0,-10,50>,$
        MEAS/XYZ,<0,-10,50>,$
        NOM/IJK,<0,0,1>,$
        MEAS/IJK,<0,0,1>
```

Measure features here

OP1_AEROFOIL_ALIGNED=ALIGNMENT/START,RECALL:OP1_ROT_LINE_ALN,LIST=YES

ALIGNMENT/LEVEL,ZPLUS, your level feature

ALIGNMENT/ROTATE,XPLUS,TO, your rotate feature ,ABOUT,ZPLUS

ALIGNMENT/TRANS,XAXIS, your X origin feature

ALIGNMENT/TRANS,XAXIS, your Y origin feature

ALIGNMENT/TRANS,XAXIS, your Z origin feature

ALIGNMENT/TRANS_OFFSET,ZAXIS, value

ALIGNMENT/TRANS_OFFSET,ZAXIS, value

ALIGNMENT/TRANS_OFFSET,ZAXIS, value

ALIGNMENT/ROTATE_OFFSET, value,ABOUT,XPLUS

ALIGNMENT/ROTATE_OFFSET, value,ABOUT,YPLUS

ALIGNMENT/ROTATE_OFFSET, value ,ABOUT,ZPLUS

ALIGNMENT/END

SAVE/ALIGNMENT,JJ115234_S_R_R.aln,MACHINETOPARTS

DISPLAYPRECISION/3

DIM TEST_P1_OP50= LOCATION OF POINT OP1_TEST1 UNITS=MM ,$

GRAPH=OFF TEXT=OFF MULT=10.00 OUTPUT=BOTH HALF ANGLE=NO

AX	NOMINAL	+TOL	-TOL	MEAS	DEV	OUTTOL	
X	0.000	0.000	0.000	0.002	0.002	0.002	-->
Y	0.000	0.000	0.000	-0.001	-0.001	0.001	<--
Z	0.000	0.000	0.000	0.001	0.001	0.001	-->

END OF DIMENSION TEST_P1_OP50

DIM TEST_P2_OP50= LOCATION OF POINT OP1_TEST2 UNITS=MM ,$

GRAPH=OFF TEXT=OFF MULT=10.00 OUTPUT=BOTH HALF ANGLE=NO

AX	NOMINAL	+TOL	-TOL	MEAS	DEV	OUTTOL	
X	0.000	0.000	0.000	-0.001	-0.001	0.001	<--
Y	-10.000	0.000	0.000	-9.999	0.001	0.001	-->
Z	50.000	0.000	0.000	50.001	0.001	0.001	-->

END OF DIMENSION TEST_P2_OP50

IF_GOTO/ABS(TEST_P1_OP50.X.DEV)>0.003,GOTO = OP1_START_ALN

IF_GOTO/ABS(TEST_P1_OP50.Y.DEV)>0.003,GOTO = OP1_START_ALN

IF_GOTO/ABS(TEST_P1_OP50.Z.DEV)>0.003,GOTO = OP1_START_ALN

IF_GOTO/ABS(TEST_P2_OP50.X.DEV)>0.003,GOTO = OP1_START_ALN

IF_GOTO/ABS(TEST_P2_OP50.Y.DEV)>0.003,GOTO = OP1_START_ALN

IF_GOTO/ABS(TEST_P2_OP50.Z.DEV)>0.003,GOTO = OP1_START_ALN