PROPOSALS FOR KECK TELESCOPE POINTING ALGORITHMS

P. T. Wallace[*]

22nd December 1994

[*]Starlink Project, Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire OX11 0QX, United Kingdom. E-mail p.t.wallace@rutherford.ac.uk.

Contents

1 INTRODUCTION

This report proposes algorithms for pointing and tracking an altazimuth telescope to the comparatively high accuracies specified for the 10-meter telescope of the W.M.Keck Observatory. The relevant standard timescales and reference frames are discussed, together with recipes for achieving the required transformations; various telescope pointing corrections are also taken into account. Vector and matrix methods are used throughout, and there is considerable emphasis on rigor. However, efficiency considerations are not overlooked, and a method of minimising the consumption of computer time is described in detail.

I refer throughout to certain items of software (in particular the coordinate conversion utility COCO and the library SLALIB) written by me and distributed by the UK Starlink Project. This software is available free to all astronomical research organizations, as part of normal scientific exchange. The routines are all written in VAX Fortran. The departures from the ANSI Fortran 77 standard are not major (confined almost entirely to the US Department of Defense extensions), and portability to other machines has been kept in mind. Indeed, strictly Fortran 77 versions of COCO, including all the SLALIB routines it uses, have been produced without difficulty for Perkin-Elmer and Facom machines. COCO contains a prescription for most of the star coordinate transformations that will be needed to control the Keck Telescope, while the SLALIB subprograms may be used directly, or cannibalized, or used as specifications for a rewrite in another language.

2 GENERAL STRATEGY

2.1 Basic Transformation Flow

Figure 1, below, identifies the main steps in transforming a position in the user's chosen coordinate system (called 'tracking coordinates') into one that the servo software can directly compare with the (corrected) azimuth and elevation encoder readings. At this stage nothing is being said about which calculations should be performed at what frequency, except (i) that the simplest approach would be to execute the whole sequence at servo rates and (ii) some degree of interpolation is nevertheless likely to be used to save CPU time (the precession matrix need not be recalculated at the full rate for example).

In Figure 1, the coordinates marked $\rightarrow$ are those appropriate for applications software to supply as demands, including scan patterns and adjustment via pushbuttons etc, to start the sequence of pointing calculations. It will be necessary in different cases to perform some preprocessing before target coordinate data are ready for use as tracking coordinates; for example it is appropriate to correct a mean $[\alpha, \delta]$ for space motion and parallax before use as a demand because these apply to the star itself rather than to the reference frame.

The suggested update rate of 20 Hz is fast enough to give the appearance of instantaneous response to pushbutton demands and is unlikely to limit appreciably the dynamic response available to programs which coordinate telescope movement with data acquisition. Even 10 Hz may prove to be adequate in the unlikely event that CPU economy is an issue. For simplicity, the pointing update interval rate should be an integer multiple of the servo update rate: 20/20, 10/30, 20/40 etc.

I propose that an approach be adopted where all the positional astronomy is done with complete rigor as far as is reasonably possible. Such a policy (a) will not avoidably erode the error budget and (b) will facilitate comparison with astrometric software available elsewhere. Where excessive cost – for example in CPU time – can be demonstrated, this policy can of course be relaxed.

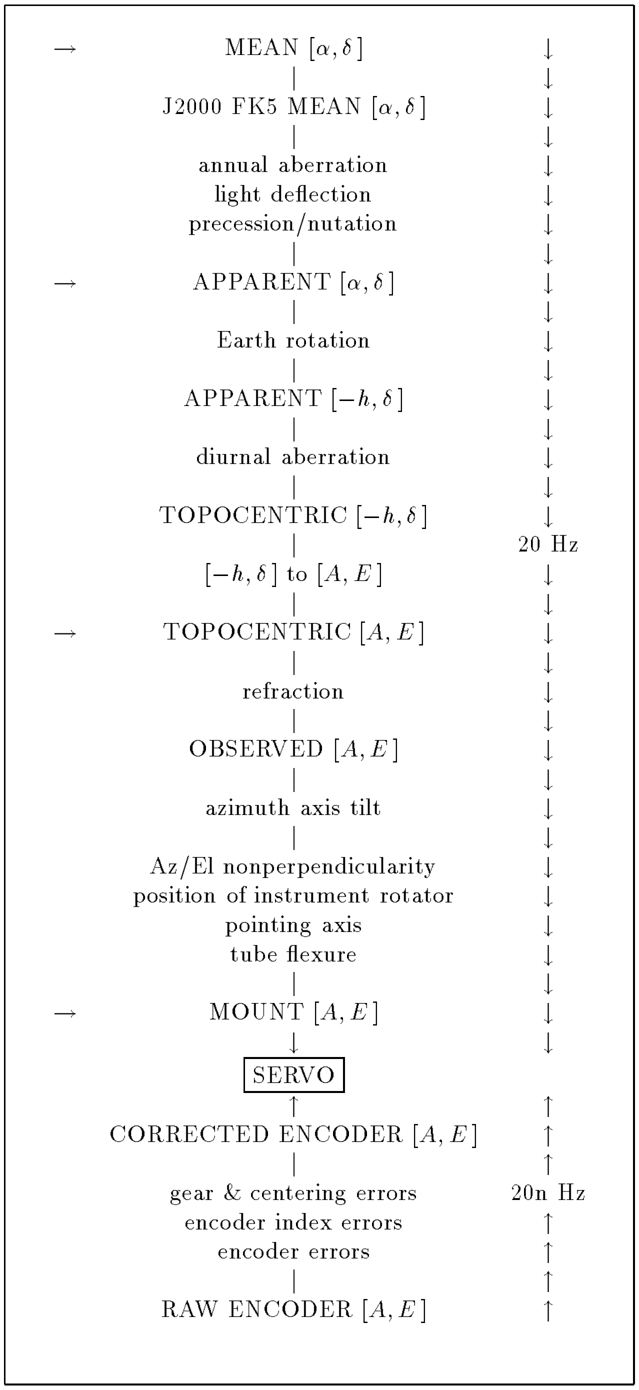

Figure 1: The pointing flow

The set of transformations shown describes the relationship between the target position (one of those marked →) and the required telescope encoder readings. There are two major transformations: $[\alpha, \delta]$ to $[-h, \delta]$, and $[-h, \delta]$ to azimuth and elevation $[A, E]$. The others are all minor.

2.2 The Virtual Telescope

The tracking coordinates are an interface to the virtual telescope, a simulation of an ideal device produced by obscuring the defects of the real telescope with layers of software. The virtual telescope is generally the only one that should interest the astronomer, but there will be some respects in which display or control of the real device is required. For example:

When a new target is presented, the virtual telescope will be in position instantaneously, whereas the real telescope will clearly take time slewing and settling before observing can begin. Both the astronomer and automated data acquisition systems will need to know the truth. In the slewing case, the astronomer will probably be satisfied simply by looking at a readout of mount $[A, E]$, but might also like a readout of 'distance to go' or a picture of the telescope orientation. The servo software could produce an 'in position and tracking' status once both the position and velocity error are within acceptable bounds (there could be several grades, or one that can be adjusted to match the seeing). The status should be accessible to instrument computers (which as well as knowing when to start an exposure might be able to close the shutter during wind gusts) and should also switch on a large green light to inform the astronomer.

Feedback on servo performance will be very useful if displayed in a vivid way which accurately conveys the dynamic nature of the information concerned. I feel sure this has to mean rapidly updated graphic displays rather than numbers, and suggest that the instantaneous position errors in the two axes are displayed as X and Y in real time on a CRT and are also plotted against time on pseudo strip-chart displays, one per axis. The CRT display, which would be controlled via DACs by the servo software (at the full servo rate) could be switchable between $(\delta A, \delta E)$ and $(\delta A \cos E, \delta E)$ and might even have quasi-logarithmic scaling to increase the dynamic range. Wind gusts should be very apparent on such a display, and any rapid oscillations. The strip-chart display (which could continuously represent the last few minutes of tracking) would also show interesting features of the wind gusting and other disturbances.

Cable wraps and mechanical interference phenomena will affect slewing strategy – when should you go the long way round? I suspect it is enough to provide all the relevant information to the operator, preferably using graphics displays, and to let him decide. Even if the decision can be made automatically with reasonable reliability there should be a manual override.

2.3 Base and Offset from Base

I propose that the Keck control system follows the AAT in allowing application programs which move the telescope to specify the tracking position (one of the ones marked $\rightarrow$ in Figure 1) as a base (two numbers, e.g. α and δ) and an offset from base (a further two numbers, e.g. $\Delta\alpha$ and $\Delta\delta$). Although the tracking loop has only to add the two pairs together, which the application software could equally well do, having the mechanism available as part of the tracking system relieves the application software of remembering where it started (useful when cleaning up after an abort) and also helps the astronomer understand what is going on during a complex scanning or offsetting maneuver.

A base and offset concept is also valuable when specifying the pointing axis position and the collimation corrections.

3 REFERENCE FRAMES

3.1 Introduction

In general the coordinates of the target object will need to be converted, before use, from the form in which they were entered to the form required to begin the pointing flow. Two reference frames (or, loosely, coordinate systems) will thus need to be allowed for simultaneously: 'target coordinates', in which the position of the target object is supplied, and 'tracking coordinates', which start the pointing flow. The required conversions are included in the repertoire of the Starlink COCO program. It is probably worth doing full COCO-style transformations (when the telescope is commanded to move to the target) even though this may appear to be excessively fussy. Certainly offsetting from nearby bright stars will be more assured if effects like parallax have been allowed for.

Note that distinguishing between the target and tracking reference frames shows why the corrections for space motion and parallax are not part of the pointing flow: they are properties of the source and not of the reference frame. Were they to be included in the pointing flow and the telescope be manually guided from the original target star to one in the field, then the indicated $[\alpha, \delta]$ would only be correct if the field star happened to share the target's proper motions, radial velocity, and parallax, which is unlikely. Thus the $[\alpha, \delta]$ which starts off the flow is the position at the current epoch in the nominated tracking reference frame.

Astronomers will want to work in several different reference frames, in some cases without appreciating the subtlety of what they are doing. For example, an astronomer who, following some published recipe, first points the telescope at the B1950 $[\alpha, \delta]$ of the Crab Pulsar and then moves 300 arcsec due North to a particular spot in the Nebula rightly expects the RA to stay fixed and the Dec to change by 300 arcsec; if, however, he were to enter the position in J2000 coordinates instead and then offset by the same amount, he would discover that the telescope was positioned over 1 arcsec from the correct point due to the relative rotation between the B1950 and J2000 systems. The consequence of this is that the pointing flow has to start in the user's preferred coordinate system and cannot, for example, always be apparent $[\alpha, \delta]$. This alone considerably increases the amount of computation that must be done during tracking.

Obvious tracking reference frames to consider include not only equatorial and altazimuth coordinates but also ecliptic, galactic, etc. However, I doubt whether the latter options are really useful, and suggest that they are left out of the initial system. (But they may be required for information displays and logging.) The required transformations are all in COCO's repertoire and there is no problem in providing them if users really want them. It is possible some radioastronomers might want an exotic flavor of mean $[\alpha, \delta]$ where the reference frame is the old pre- IAU 1976 one but the E-terms of aberration are not included. Again, COCO specifies how to do this.

Summarizing, I recommend that the telescope control system should both (i) accept target positions and (ii) control the telescope in at least the following coordinate systems:

- Mean $[\alpha, \delta]$, old style (i.e. before the IAU 1976 resolutions, loosely called FK4 and frequently referred to the mean equator and equinox of Besselian epoch 1950.0 – hence B1950), of any equinox.
- Mean $[\alpha, \delta]$, new style (i.e. after the IAU 1976 resolutions, loosely called FK5 and frequently referred to the mean equator and equinox of Julian epoch 2000.0 – hence J2000), of any equinox.
- 'Local' apparent $[\alpha, \delta]$, new style. This would be the form output by planetary ephemeris programs, which would allow for parallax (geocentric and topocentric) and planetary aberration, and would constantly update the demand $[\alpha, \delta]$ to track the object (all of which are essential for the Moon and at least desirable for the planets).
- Topocentric $[A, E]$. This would be useful occasionally where an application wanted to do all the other transformations itself – satellite tracking would be an example.
- Mount $[A, E]$. The obvious applications are engineering ones – parking the telescope for example.

Both the target and tracking reference frames should default to FK5 J2000.

When entering target data, there could either be separate commands for specifying the coordinate system and for entering coordinates, or the target coordinate system could be specified by parameters supplied along with the coordinates. What is most convenient depends on the syntax of the command language, and the facilities for parameter defaulting etc.

3.2 Some Remarks on Target Mean RA/Dec Data Entry

The way input coordinates are supplied by the user depends heavily on the nature of the command language that the telescope control system will support. However, I have some general advice concerning handling mean $[\alpha, \delta]$ target positions, which will be by far the most common sort.

As mentioned in the previous section, two styles of mean $[\alpha, \delta]$, which I will call FK4 and FK5, will need to be supported. Very often the astronomer will not be aware of the subtle differences between them (which can lead to mistakes of up to an arcsecond) and the commands he uses will need to have helpful defaulting rules so that the difficulties are masked and he gets the right result without understanding the fine details.

Both sorts of mean $[\alpha, \delta]$ require the following data if they are to be completely specified:

- The $[\alpha, \delta]$ position itself
- Whether it is in the old FK4 system or the new FK5 system
- The equinox ('epoch of mean equator and equinox' in full)
- The epoch (time zero for the proper motion correction)
- Proper motion
- Parallax
- Radial velocity

The commands for specifying the target reference frame and for entering target coordinates should be so designed that most target stars will only have to be entered as a plain $[\alpha, \delta]$.

Depending on the properties of the chosen command language, I suggest defaulting conventions along the following lines:

- The equinox (for example B1950) can be used to imply the system, with prefix B meaning 'old system' or 'FK4', and prefix J meaning 'new system' or FK5. If no prefix is specified, the system can be reliably deduced from the value supplied, so that an equinox before 1984.0 has an implied B prefix, and 1984.0 or later implies J. The equinox and system together should initially default to J2000 FK5.
- The epoch, which determines the amount of proper motion to allow for, will generally be supplied as a year (e.g. 1976.44) but could also be accepted as year, month, day. If a year, for formality's sake a B or J prefix could be used as for the equinox, though this will have a negligible effect on the result. The value should default to that of the equinox, which is almost always the case in star catalogues. It would be wise *not* to allow the epoch to default in the case of FK4 coordinates where the proper motion has not been supplied and is presumed inertially zero. (An object such as a QSO has a fictitious non-zero proper motion in the FK4 system, which is not an inertial frame. This is not well known to observational astronomers and is one of two celebrated sources of confusion, the other being the presence in pre- IAU 1976 mean places of the E-terms of aberration.)
- Proper motions (which must be supplied as a pair) default to zero in the new system and to inertially zero in the old system.
- The parallax and radial velocity both default to zero.

Most celestial targets will be $[\alpha, \delta]$, with (inertially) zero proper motions, in either B1950 coordinates at some specified epoch, or J2000. For pointing calibration stars it will be important to include proper motions and in some cases parallax and radial velocity.

3.3 Transformation of Mean Places

In this section I will list all the steps required to perform the following transformations:

- Target mean $[\alpha, \delta]$ (three sorts) to tracking mean $[\alpha, \delta]$ (two sorts).
- Tracking mean $[\alpha, \delta]$ (two sorts) to apparent.

The three sorts of target mean $[\alpha, \delta]$ are:

- Old style (FK4) with known proper motion in the FK4 system, and with parallax and radial velocity either known or assumed zero.
- Old style (FK4) with inertially zero proper motion, and with parallax and radial velocity assumed zero.
- New style (FK5) with proper motion, parallax and radial velocity either known or assumed zero.

The two sorts of tracking mean $[\alpha, \delta]$ are:

- Old style (FK4).
- New style (FK5).

(The procedures to be described attempt to reduce program size and to improve modularity by performing all conversions via one standard reference frame, namely J2000 FK5. Though in many cases it would be possible to devise a specialized routine for each combination of target reference frame and tracking frame, or even to transform optimally all the way to $[A, E]$, the software will be easier to maintain and enhance if an indirect modular approach is taken.)

We can thus construct any of the required transformations out of a total of seven building blocks, most of which need only be executed once when acquisition of the target is requested. They are as follows.

Required once only, at the start of a new track, one of:

a) FK4 with proper motion to J2000 FK5 current epoch

b) FK4 with no proper motion to J2000 FK5 current epoch

c) FK5 to J2000 FK5 current epoch

followed by one of:

d) J2000 FK5 to FK4

e) J2000 FK5 to FK5

Required continuously during tracking, after adding the offsets from base, one of:

f) FK4 to J2000 FK5

g) FK5 to J2000 FK5

As an example, consider the case where the target has been specified in 1900 coordinates, proper motions have been given, and the telescope is being controlled in 1950 coordinates. The required procedures would be (a) then (d) before tracking, and (f) continuously thereafter.

The steps comprising each building block are given in the following sections. Each step requires one Starlink SLALIB call (name in parentheses) or equivalent code. A summary diagram is given later, including the mean to apparent stage.

a) FK4 with proper motion to J2000 (once only)

1. Space motion to the current epoch. (PM)
2. Remove E-terms of aberration. (SUBET)
3. Precess to B1950. (PRECES)
4. Add E-terms. (ADDET)
5. Transform to J2000, no proper motion. (FK45Z)
6. Parallax. (See MAPQK)

b) FK4 without proper motion to J2000 (once only)

1. Remove E-terms. (SUBET)
2. Precess to B1950. (PRECES)
3. Add E-terms. (ADDET)
4. Transform to J2000, no proper motion. (FK45Z)

c) FK5 to J2000 (once only)

1. Space motion to the current epoch. (PM)
2. Precess to J2000. (PRECES)
3. Parallax. (See MAPQK)

d) J2000 to FK4 (once only)

1. Transform to B1950, no proper motion. (FK54Z)
2. Remove E-terms. (SUBET)
3. Precess to final equinox. (PRECES)
4. Add E-terms. (ADDET)

e) J2000 to FK5 (once only)

1. Precess to final equinox. (PRECES)

f) FK4 to J2000 (tracking)

1. Remove E-terms. (SUBET)
2. Precess to B1950. (PRECES)
3. Add E-terms. (ADDET)
4. Transform to J2000, no proper motion. (FK45Z)

g) FK5 to J2000 (tracking)

1. Precess to J2000. (PRECES)

These pathways are presented diagrammatically in Figure 2.

There is obviously scope for optimization and approximation in the above procedures, if necessary. An optimization would be to omit redundant steps whenever the target is specified in B1950 or J2000 coordinates, or when the telescope is being controlled in J2000 coordinates. Another would be to avoid multiple conversions between spherical and Cartesian coordinates by staying in x, y, z. A simplification would be to omit the pairs of steps which first subtract and then add back the E-terms, which though rigorously correct will have only an inconsequential effect on the result for the range of equinoxes that will be used. However, I feel that the disadvantages of these changes marginally outweigh the advantages, and I recommend the transformations are implemented exactly as given unless other practical issues emerge.

3.4 Mean to Apparent

Transformation from J2000, FK5, current epoch, to apparent place, required either continuously during tracking (where the telescope is being controlled in mean place, the normal case) or just once at the start of a new track (in the rare case where the target has been specified as a mean place and the telescope is being controlled in apparent place) involves the following effects:

- Light deflection – the gravitational lens effect of the sun.
- Annual aberration.
- Precession/nutation.

Though the light deflection is significant at the limb of the Sun (1.74 arcsec) it falls off rapidly and has shrunk to about 0.02 arcsec at an elongation of 20° from the Sun, which is presumably closer than will ever be used. The effect is thus negligible for our purpose and could be omitted. However, if there is no CPU time problem it may be best to perform the correction for the usual reasons of (a) rigor and (b) to assist comparison with other software.

The annual aberration is a function of the Earth's velocity relative to the solar system barycenter (available through the Starlink SLALIB routine EVP) and produces shifts of up to about 20.5 arcsec.

The precession/nutation, from J2000 to the current epoch, is expressed by a rotation matrix which is available through the Starlink SLALIB routine PRENUT.

The whole transformation can be done using the Starlink SLALIB routine MAP, with the proper motions, radial velocity, and parallax all set to zero, and the equinox to 2000. This is, however, a wasteful approach as both the Earth velocity and the precession/nutation matrix can be calculated relatively infrequently without ill effect. A more efficient method is to precompute the target-independent parameters with the MAPPA routine and then to use MAPQKZ.

4 TIMESCALES AND POLAR MOTION

The following three timescales are required for telescope pointing and other observatory purposes:

- UTC (coordinated universal time) is needed for general logging.
- ST (local apparent sidereal time) is needed for the Earth rotation part of the telescope pointing flow (and astronomers will expect to see it displayed on a VDU somewhere even though all they will use it for is to do mental calculations about source rise/set times, something arguably better handled by graphics displays provided by the computer).
- TDB (barycentric dynamical time) is needed for various dynamical calculations (e.g. planetary predictions) and should also be provided to the instrument computers for timing variable sources.

To obtain these forms of time, the following are needed:

- UTC. This will come from the observatory time service and will be assumed to include leap seconds (see later).
- The telescope mean longitude and latitude.
- International Earth Rotation Service (IERS) *Bulletin A*, which gives information about leap seconds ($\Delta TT \equiv TT - UTC$) and Earth orientation ($\Delta UT \equiv UT - UTC$ and polar motion).

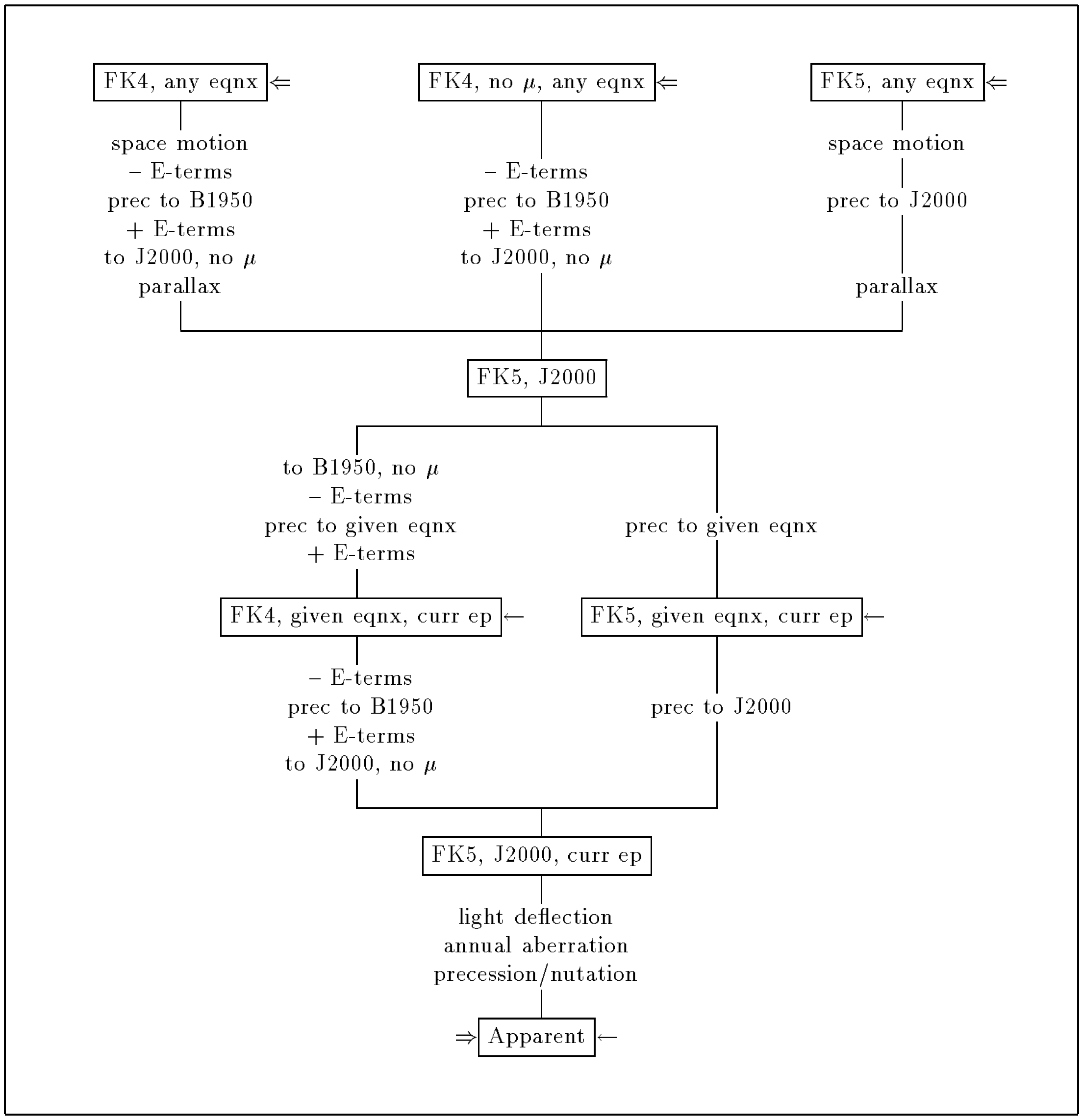

Figure 2: Transformations for mean $[\alpha, \delta]$

The forms marked ⇒ are those available for target data entry (target coordinates), a choice of four; the forms marked → are available for telescope control (tracking coordinates). Pick one of each and follow the flow downwards. The sequences down to the chosen tracking coordinates have only to be executed once per new target, but all the transformations from that level down have to be performed at the full pointing rate.

It will be convenient to correct the telescope mean longitude and latitude for polar motion at the start of the observing session even though in principle it should be done continuously during the vector operations of the pointing flow. The computations will require the polar x, y, the telescope mean longitude, and the telescope mean geodetic latitude. The simple first order expressions on page B59 of the 1986 Astronomical Almanac are adequate.

The ΔUT given in the bulletins will not be up to date and will have to be extrapolated to give a new value every day. A simple linear extrapolation will be just good enough for pointing the telescope; more sophisticated and accurate extrapolations exist which take into account known seasonal effects. An alternative is daily access to the USNO dial-up time service.

TDB can be determined as follows:

1. To the UTC add ΔTT giving terrestrial time TT (or add ΔAT to UTC giving international atomic time TAI and add 32.184sec giving TT).
2. Using the Starlink SLALIB routine RCC, add the small corrections for gravitational redshift and transverse Doppler effect (peak to peak 3.3 ms, clearly of no significance for telescope pointing but probably worth doing partly for rigor but mainly as a service to data acquisition applications involving timing of variable sources).

There are several forms of sidereal time; we will use the name ST to mean the local apparent sidereal time computed using a telescope longitude which has been corrected for polar motion. The steps required for computing ST are as follows:

1. Compute UT by adding ΔUT to the UTC.
2. Compute the Greenwich mean sidereal time GMST from the equation on page B6 of the 1986 Astronomical Almanac. This is implemented in the Starlink SLALIB routine GMST.
3. Add the equation of the equinoxes, giving the Greenwich apparent sidereal time GAST. The equation of the equinoxes can be obtained by means of the Starlink SLALIB function EQEQX, or preferably computed along with the nutation matrix via the routines NUTC and NUTM. The time argument for either EQEQX or NUTC is TDB, which will be available and should be used even though for this purpose UTC is commonly used directly without appreciable loss of accuracy.
4. Add the telescope East longitude to give ST, the local apparent sidereal time.

It will not, of course, be necessary to go through the whole of the above calculation continuously as part of the telescope tracking. For example, the sidereal time could be implemented within the telescope control computer as three double precision numbers:

- `STRAW` raw uncorrected sidereal time
- `STCOR` correction from raw ST to LAST
- `ST` true local apparent sidereal time

and, assuming the tracking loop is executed continuously whatever the telescope is doing, the following would be done each time through (assuming 20 Hz UTC operation for the sake of argument):

```
      DOUBLE PRECISION FREQ,DST
      PARAMETER (FREQ=20D0,
     :           DST=1.00273790934D0/FREQ)
          :
```

```
      :
STRAW = STRAW + DST
ST = STRAW + STCOR
      :
```

Immediately on startup, and every few minutes or so thereafter, the full ST calculation given earlier should be carried out to determine `STCOR`.

(The above algorithm relies on never missing a 20 Hz update or being falsely triggered. A slightly more complicated algorithm which is always sensitive to the absolute UTC would be better.)

There are realtime aspects that need attention to ensure consistent UTC and ST. One simple technique might be:

1. Grab UTC etc, and `STRAW`.
2. Look at UTC and `STRAW` again; if either has changed, they might be inconsistent, so go back to step 1.
3. Compute correct sidereal time `STC` from UTC etc.
4. Set `STCOR` to `STC-STRAW`.

Leap seconds, which happen at 0 hours UT on January 1 or July 1, are announced several months in advance in time bulletins. Three actions will be required:

1. When the announcement is first made, a computer file accessible to the telescope control software can be updated to show the date at which the leap second will occur and the ΔTT after that point. The Starlink SLALIB routine DTT is a guide to what is needed; it is not ideal for direct use as it requires either the use of a shared library or for all applications software to be relinked each time a new leap second is announced.
2. During the day preceding the 0 hours UTC at which the leap second is to occur, the leap second switch on the timecode generator must be armed.
3. A compensating 1 second step must be introduced into the ΔUT. For example, if the leap second is a positive one (so that the UTC went 23:59:59, :60, :00,) the ΔUT, which will have been, say, -853 ms, will become $+147$ ms. (The purpose of leap seconds is, of course, to keep ΔUT within about ±1 sec, so that UTC can be used directly for astronavigation and other relatively low precision applications.)

5 POINTING TERMS

5.1 Vector Methods

I will specify all the telescope pointing calculations using Cartesian ($\equiv$ rectangular $\equiv$ direction cosines $\equiv x, y, z$) unit vectors rather than old fashioned spherical trigonometry methods. The now widely used vector methods give greater protection against pole problems, more rarely require departures from rigor, maintain more uniform accuracy over the celestial sphere, and allow more succinct expression.

The coordinate convention I will use is as follows. Spherical coordinates A,B are such that B is $\pm\pi/2$ at the poles, and A is positive anticlockwise as seen from the positive B pole. $[\alpha, \delta]$ conform to this convention, and longitude/latitude if longitude is measured East, but not $[h, \delta]$ or Keck $[A, E]$. (I shall be using $[-h, \delta]$ internally, and A measured from South through East.) The corresponding Cartesian coordinates have the x-axis through the point $A = 0, B = 0$, the z-axis at $B = +\pi/2$, and the y-axis at $A = +\pi/2, B = 0$.

The procedures for conversion between spherical and Cartesian coordinates can easily be deduced from the above and are well known (for example see the Starlink SLALIB routines DCSC and DCCS).

Rotations of the reference frame are produced by multiplying the x, y, z column vector by a 3×3 orthogonal matrix (a tensor of Rank 2, or *dyadic*), where the three rows are the vectors in the old coordinate system of the three new axes.

Shifts of the direction of a vector need careful handling if the vector is to remain of length unity, an advisable precaution so the x, y, z components are always available to mean the cosines of the angles between the vector and the axis concerned. The telescope pointing calculations will have two types of shifts to deal with, one where a small vector of arbitrary direction is added to the unit vector, and one where there is a displacement in elevation alone.

For a shift produced by adding a small x, y, z vector $\mathbf{D}$ to a unit vector $\mathbf{V1}$, the resulting vector $\mathbf{V2}$ has direction $< \mathbf{V1} + \mathbf{D} >$ but is no longer of unit length. A better approximation is available if the result is multiplied by a scaling factor of $(1 - \mathbf{D} \cdot \mathbf{V1})$, where the dot means scalar product. In Fortran:

```
F = (1D0-(DX*V1X+DY*V1Y+DZ*V1Z))
V2X = F*(V1X+DX)
V2Y = F*(V1Y+DY)
V2Z = F*(V1Z+DZ)
```

The correction for diurnal aberration is an example of this type of shift.

When a small change in elevation δE is made to a direction vector (for example in the case of refraction), the direction of the result can be obtained by making the allowable approximation $\tan\delta E \approx \delta E$ and adding a adjustment vector of length δE normal to the direction vector in the vertical plane containing the direction vector. The z-component of the adjustment vector is $\delta E \cos E$, and the horizontal component is $\delta E \sin E$ which has then to be resolved into x and y in proportion to their current sizes. To approximate a unit vector more closely, a correction factor of $\cos\delta E$ can then be applied, which is nearly $(1 - \delta E^2/2)$ for small δE. Expressed in Fortran, for initial vector `V1X,V1Y,V1Z`, change in elevation `DEL` (+ve $\equiv$ upwards), and result vector `V2X,V2Y,V2Z`:

```
COSDEL = 1D0-DEL*DEL/2D0
R1 = SQRT(V1X*V1X+V1Y*V1Y)
F = COSDEL*(R1-DEL*V1Z)/R1
V2X = F*V1X
V2Y = F*V1Y
V2Z = COSDEL*(V1Z+DEL*R1)
```

Note that the division by `R1` gives a zenith problem. This is unlikely to be a serious difficulty as long as the effect concerned is zero at the zenith (which is true of refraction but not collimation) and of a functional form that allows the equations to be simplified. The refraction algorithm is well behaved in this respect; there is an overall $\cot E$ which allows the `R1` to be cancelled to give $\csc E$, and hence no problems until the horizon.

5.2 Earth Rotation

This is the first of two major rotations of the reference frame, the part of the pointing flow which converts Right Ascension to minus Hour Angle. (As already explained, it is more convenient to use minus HA than to have a lefthanded coordinate system.) It requires the local apparent sidereal time, the derivation of which was covered earlier. The transformation can be expressed as the following orthogonal matrix:

$$\begin{bmatrix} +C & +S & 0 \\ -S & +C & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The symbol C represents the cosine of the local apparent sidereal time (24 hours = 2π radians), and the symbol S is the sine. The multiplications can be written down explicitly as follows (in Fortran):

```
X2 = +C*X1+S*Y1
Y2 = -S*X1+C*Y1
Z2 = Z1
```

`X1,Y1,Z1` is the $[\alpha, \delta]$ vector and `X2,Y2,Z2` the apparent $[-h, \delta]$ vector.

5.3 Diurnal Aberration

This is the component of aberration due to the motion of the observatory around the Earth's axis, and causes a shift in the apparent direction of the target which, at Mauna Kea, will be up to about 0.3 arcsec (for targets on the meridian), the ratio between the rotational speed of the observatory as the Earth spins and the speed of light. Starting from Cartesian coordinates `X1,Y1,Z1` in the local $[-h, \delta]$ system, and an aberration constant for Mauna Kea `DIURAB`, allowance for diurnal aberration can be made as follows (in Fortran):

```
F = (1D0-DIURAB*Y1)
X2 = F*X1
Y2 = F*(Y1+DIURAB)
Z2 = F*Z1
```

`X1,Y1,Z1` is the apparent $[-h, \delta]$ vector and `X2,Y2,Z2` the topocentric vector $[-h, \delta]$ vector.

`DIURAB` is simply the speed of rotation (sidereal) of the observatory, in units of c. It is proportional to the distance of the observatory from the Earth's spin axis, which can be obtained by means of the Starlink SLALIB routine GEOC. If this distance is in AU, the multiplier is:

$$2\pi/(0.99726956634 \times 173.14463331)$$

5.4 -HA/Dec to Az/El

This is the second of the two major rotations of the reference frame, from equator based coordinates to horizon based coordinates. The rotation is about the y-axis in the $[-h, \delta]$ system, so that the z-axis moves from the North celestial pole to the local zenith at the telescope. This rotation is through 90° minus the astronomical latitude, corrected for polar motion. (The polar motion correction is described in the section on timescales, earlier.) The difference between the astronomical latitude (which is related to the direction of the local gravity vector and is strictly needed for this transformation because the refraction and tube flexure effects are centered on the astronomical zenith) and the geodetic latitude (which is geometrical) can probably be neglected. The transformation can be expressed as the following orthogonal matrix:

$$\begin{bmatrix} +S & 0 & -C \\ 0 & +1 & 0 \\ +C & 0 & +S \end{bmatrix}$$

The symbol C represents the cosine of the telescope latitude, and the symbol S is the sine. The multiplications can be written down explicitly as follows (in Fortran):

```
X2 = +S*X1-C*Z1
Y2 = +Y1
Z2 = +C*X1+S*Z1
```

`X1,Y1,Z1` is the topocentric $[-h, \delta]$ vector and `X2,Y2,Z2` the topocentric $[A, E]$ vector.

Note that my azimuth convention is different from the one generally used as well as from the Keck Telescope's. Mine is zero in the South and $+90°$ in the East, rather than North through East. This is to preserve the handedness of the coordinate systems and hence the procedures for conversion between spherical and Cartesian coordinates. Transformation to the convention used on the Keck Telescope (simply a sign reversal in x) can take place as the final step in the pointing flow.

5.5 Refraction

The effect of atmospheric refraction is to increase the observed elevation of an astronomical object by an amount which is usually modeled as:

$$\zeta_{vac} \approx \zeta_{obs} + A \tan \zeta_{obs} + B \tan^3 \zeta_{obs}$$

where ζ_{vac} is the topocentric zenith distance (i.e. *in vacuo*), ζ_{obs} is the observed zenith distance (i.e. affected by refraction), and A and B are parameters which depend on local meteorological conditions and the effective color of the source/detector combination.

For typical observing conditions at the Keck Telescope, A will be approximately $+36$ arcsec and B approximately -0.04 arcsec.

The constant A depends most strongly on the refractive index n of the air near the telescope (A is approximately $n - 1$ radians) which can readily be computed as a function of temperature, pressure, humidity and wavelength (from formulae in Astrophysical Quantities by C.W.Allen, and elsewhere). However A also depends to some extent, and B to a large extent, on the large scale structure of the atmosphere above the telescope – the temperature and water vapor distribution with height in particular – and accurate prediction of A and B is not especially easy or fast, requiring numerical integrations. The Starlink SLALIB routine REFCO computes A and B by calling a routine REFRO (which is a repackaged form of a definitive routine developed by the RGO Nautical Almanac Office). The required input parameters for REFCO are as follows:

- temperature, pressure and relative humidity
- temperature lapse rate in the troposphere
- latitude and height
- effective wavelength

Summarizing so far, the refraction calculations will be driven by the two constants, A and B, which can be computed at startup and every few minutes thereafter, by the Starlink SLALIB routine REFCO.

Whether the meteorological parameters are read in automatically or entered by the operator depends on the reliability of the transducers; automatic updating might introduce small glitches in the tracking.

The next problem is that the above refraction formula predicts the *in vacuo* zenith distance given the refracted zenith distance, and we want to go the other way. The naive approach of simply interchanging ζ_{vac} and ζ_{obs} and reversing the sign, though approximately correct, gives avoidable errors which are just significant; for example at 20° elevation the error is about 0.14 arcsec. It is, however, possible to employ one iteration of the Newton-Raphson method to give a much better result at little extra cost. The formula is:

$$\zeta_{obs} \approx \zeta_{vac} - \frac{A \tan\zeta_{vac} + B \tan^3\zeta_{vac}}{1 + (A + 3B\tan^2\zeta_{vac})\sec^2\zeta_{vac}}$$

At 20° elevation the error is less than 0.0002 arcsec.

When the vector formulae for refraction are set down, considerable simplification is possible, a byproduct of which is the elimination of any problem at the zenith. The following Fortran procedure takes an *in vacuo* position $[A, E]$ vector `X1,Y1,Z1` and calculates the refracted position `X2,Y2,Z2`, the new coordinate system being observed $[A, E]$. The refraction constants A and B are assumed to be known:

```
ZSQ = Z1*Z1
RSQ = X1*X1+Y1*Y1
R = SQRT(RSQ)
WB = B*RSQ/ZSQ
WT = (A+WB)/(1D0+(A+3D0*WB)/ZSQ)
D = WT*R/Z
CD = 1D0-D*D/2D0
F = CD*(1D0-WT)
X2 = X1*F
Y2 = Y1*F
Z2 = CD*(Z1+D*R)
```

The real implementation may need some protection against divide by zero at the horizon. `D` is the change in elevation. The above algorithm is implemented in the Starlink SLALIB routine REFV.

5.6 Tilt of the Azimuth Axis

The surveying techniques used to set up the azimuth bearing will mean that the azimuth axis will be parallel to the astronomical vertical (the direction of gravity) to within a few arcseconds. If it intersects the celestial sphere AX radians South and AY radians East of the astronomical zenith, the tilt (simply a small rotation of the reference frame) can be allowed for by multiplying by the following orthogonal matrix:

$$\begin{bmatrix} +\cos AX & 0 & -\sin AX \\ -\sin AX \sin AY & +\cos AY & -\cos AX \sin AY \\ +\sin AX \cos AY & +\sin AY & +\cos AX \cos AY \end{bmatrix}$$

Using `SX`,`CX`,`SY`,`CY` to signify $\sin AX$, $\cos AX$, $\sin AY$, $\cos AY$, this can be coded in Fortran as follows:

```
X2 =  CX*X1-SX*Z1
Y2 = -SX*SY*X1+CY*Y1-CX*SY*Z1
Z2 =  SX*CY*X1+SY*Y1+CX*CY*Z1
```

The `X1,Y1,Z1` vector is in observed $[A, E]$, and the `X2,Y2,Z2` vector is in pre-collimation mount $[A, E]$. The tilt will vary due to azimuth journal irregularities, so AX and AY may be functions (implemented perhaps as harmonics or lookup tables) of the corrected encoder azimuth.

If necessary the approximations $\sin AX \approx AX$, $\cos AX \approx 1$, $\sin AY \approx AY$, $\cos AY \approx 1$ may be used, assuming AX and AY are small.

(The algorithm for an equatorial mounting would at this point require a large rotation, from $[A, E]$ to pre-collimation mount $[-h, \delta]$. The above matrix is rigorous and would simply need the correct AX and AY values to be used, close to $\phi - \pi/2$ and zero respectively, where ϕ is the latitude.)

5.7 Collimation Errors

Geometric difficulties at the zenith make it convenient to treat three different distinct pointing effects as a package:

- nonperpendicularity of azimuth and elevation axes
- position of instrument rotator
- position of nominated pointing axis

5.7.1 Az/El Nonperpendicularity

The $[A, E]$ nonperpendicularity $NPAE$ is a small angle, positive when the beam moves increasingly towards the left as the telescope moves up from the horizon, as you look at the sky.

$NPAE$ will be small, probably less than 5 arcsec, but may need dynamic corrections according to empirically determined models, due to journal irregularities for example.

5.7.2 Instrument Rotator Position

The position of the instrument rotator is described by two small angles, the horizontal collimation CA and the vertical collimation CE.

The horizontal collimation is the departure from perpendicularity of the incoming beam to the elevation axis, for a star focussed on the rotator axis. The sign convention I will use is that as the telescope moves up from the horizontal, CA is positive if the beam describes a small circle to the left of the nominal vertical as you look at the sky.

The vertical collimation is the elevation of the same beam when the mechanical elevation is zero, so the sign convention is such that a positive CE means that the beam is actually at a positive elevation when the corrected elevation encoder reading suggests that the telescope is horizontal.

The vertical collimation CE is logically distinct from the elevation encoder index error IE but is unlikely to be separable from pointing test analyses. It is advisable, however, to calibrate CE and IE separately, especially if there are rapidly varying terms in the encoder/gear corrections. One way this could be done would be to have fixed reference marks on the telescope moving parts so that a standard mechanical orientation can be repeated. Although not strictly required, it will obviously be best to arrange that the corrected $[A, E]$ encoder readings are reasonably close to the mechanical reality.

CA and CE might typically be up to 100 arcsec in size. Large and predictable corrections will be required for the shifts produced by the atmospheric dispersion compensator.

5.7.3 Nasmyth

At the Nasmyth foci, there will in general be a displacement between the rotator axis and the point where the elevation axis intersects the focal plane. This will have the effect of introducing elevation dependent variations into the collimation angles *CA* and *CE*.

Two extra terms, *NRX* and *NRY*, are required in the pointing model. Using a sign convention such that with the telescope tube horizontal, *NRX* is positive when the rotator axis is to the left of the elevation axis as seen by someone standing on the Nasmyth deck, and *NRY* is positive when the rotator axis is below the elevation axis. (For zero elevation, *NRX* would add to *CA* and *NRY* would add to *CE*.) With just one Nasmyth f/number, *NRX* and *NRY* are most conveniently measured in arcsec; were there to be more than one Nasmyth f/number, meters or millimeters would be more appropriate so that the values would be nominally the same at all f/numbers. *NRX* and *NRY* are likely to be very small – a few arcseconds at most.

The contributions *DCA* and *DCE* to the collimation from the Nasmyth rotator displacements *NRX* and *NRY* may be computed as follows. (Note that the algorithm requires an estimate of the post-collimation mount elevation; this can be the value from the previous iteration, and the first iteration of all can use any valid vector – the zenith for example. The vector `X2,Y2,Z2` is this estimated post-collimation mount $[A, E]$.)

```
*  Sine and cosine of post-collimation mount elevation
      SEL = Z2
      CEL = SQRT(X2*X2+Y2*Y2)

*  Nasmyth corrections
      DCA = +NRX*CEL+NRY*SEL
      DCE = -NRX*SEL+NRY*CEL
```

5.7.4 Coudé

At coudé, misalignments of the coudé 4, 5, 6 and 7 mirrors will require pointing corrections, each mirror needing two coefficients. This will not be discussed further here.

5.7.5 Pointing Axis

The position of the pointing axis relative to the rotator axis is defined by the rotator position angle *RPA* and the pointing axis x, y on the rotator XIM, YIM.

By 'pointing axis' I mean the nominated point in the focal plane to which the pointing refers (corrected for any distortion inherent in the telescope optics, when compared with ordinary tangent-plane projection geometry). The terms 'instrument aperture', 'beam', 'pointing origin' and 'optical axis' are sometimes used to mean the same thing. See the section on calibrating the pointing axis positions, later.

I will adopt a sign convention for *RPA*, XIM and YIM as follows, chosen for geometrical convenience rather than accordance with the mechanics. The mechanical position angle and x, y coordinates will need to be transformed into the pointing sign convention before use. This transformation (typically just a few sign changes) will be different for different focal positions; note in particular that the prime, bent Cass, Nasmyth and coudé have fields which are reversed relative to Cassegrain.

RPA is zero when, for elevations well away from the zenith, the projection on the sky of the rotator's y-axis points upwards (ignoring for the present any small corrections due to $NPAE$). *RPA* then increases from zero through $+90^\circ$ as the projection on the sky of the y-axis rotates anticlockwise.

YIM is positive when for zero RPA and elevations well away from the zenith the projection on the sky of the pointing axis is above the projection of the rotator axis.

The positive direction of XIM is such that the XIM and YIM axes have the conventional orientation as seen projected on the sky: the XIM axis is 90° clockwise of the YIM axis.

RPA (the actual rotator position angle, not the demanded one) should be stored internally in radians but talked about in degrees.

XIM and YIM should be in units of length (meters is probably the appropriate unit, or possibly millimeters) so that the same instrument used at different f/numbers will have the same offsets. However, the scale of the pointing compensation will depend on the focal length `F` of the telescope, and it will be more important to use consistent values for the focal length than to know the focal length precisely. It will be convenient if XIM and YIM are available internally in (loosely) radians:

```
XR = XIM/F
YR = YIM/F
```

5.7.6 The Combined Collimation Correction

All the collimation effects described above can be combined using essentially plane geometry (at the edge of a half degree field the departure from gnomonic geometry, for example, is considerably less than 0.1 arcsec) to yield a net pointing axis position. In Fortran, the algorithm is as follows (with `X1,Y1,Z1` the pre-collimation mount $[A, E]$):

```
*  Correct the position angle for Az/El nonperpendicularity
      RXY2 = X1*X1+Y1*Y1
      RXY = SQRT(RXY2)
      PA = RPA+NPAE*RXY
      SPA = SIN(PA)
      CPA = COS(PA)

*  Pointing axis position (on sky XI +ve right, ETA +ve up)
      XI  = +XR*CPA+YR*SPA-(CA+DCA+NPAE*Z1)
      ETA = -XR*SPA+YR*CPA+CE+DCE
```

This overall pointing axis position `XI,ETA` can now be used to derive a transformation predicting the required mount $[A, E]$ to align the pointing axis with the target. The pre-collimation mount coordinates `X1,Y1,Z1` can be regarded as the 'star' and the post-collimation mount coordinates `X2,Y2,Z2` the 'telescope':

```
*  Predict mount vector
      XI2 = XI*XI
      ETA2P1 = ETA*ETA+1D0
      SDF = Z1*SQRT(XI2+ETA2P1)
      R2 = RXY2*ETA2P1-Z1*Z1*XI2
      R = SQRT(R2)
      C = (SDF*ETA+R)/(ETA2P1*RXY*SQRT(R2+XI2))
      X2 = C*(X1*R-Y1*XI)
      Y2 = C*(Y1*R+X1*XI)
      Z2 = (SDF-ETA*R)/ETA2P1
```

(The above algorithm is a Cartesian adaptation of the Starlink SLALIB routine DTPRD.)

Any encoder zero point error is assumed to be absorbed into the *CE* coefficient.

In operational code, tests will need to be inserted for two cases which cause the above routine to deliver incorrect results. In rough terms, these cases are where the zenith distance of the target is less than (i) the distance between the rotator axis and the pointing axis, or (ii) the net horizontal collimation. Case (i) must be handled properly if the telescope tube is to be allowed to tip past the zenith (unwise if hysteresis is to be kept to a minimum). In case (ii) the target is impossible to reach.

Note that if the rotator is tracking the changes required to match the azimuth adjustment will introduce second order effects in the collimation corrections. In a precise treatment it would be necessary to iterate.

As already mentioned, for any given value of net horizontal collimation, any position on the sky less than that distance from the zenith cannot be reached, irrespective of azimuth speed performance, and the above algorithm will give arbitrary results in azimuth. If, however, the offset of the nominated pointing axis from the rotator axis exceeds the *CA* value itself, rotations of the instrument mount will always be able to achieve a net zero horizontal collimation, and if appropriate movements in azimuth are also available the zenith could be observed. This could just conceivably have a practical use: a detector placed near the edge of the usable field might be able to track a region of sky right through the zenith (not on the Keck Telescope, however – the field diameter and the maximum azimuth speed are too small). That this is possible can easily be seen by considering a widefield photographic exposure of a field whose declination is such that the zenith will pass by at the edge of the plate during tracking. Should a CCD exposure of a field on the edge of such a plate be needed instead, the CCD could be positioned where that part of the plate would have been.

5.8 Tube Flexure

The calculations so far have ignored the effects of gravity on the telescope tube. The next step is to imagine the gravity being "turned on", causing the telescope to deflect vertically and lose alignment with the target; what mount adjustment will be required to restore the alignment?

A plausible starting point for the tube flexure model is to assume that the entire assembly obeys Hooke's law, so that the pointing shift is vertical and proportional to the component of gravity normal to the tube axis. This leads to a model:

$$\zeta \approx \zeta_{tel} + TF \sin \zeta_{tel}$$

where ζ is the 'observed' zenith distance for the nominal tube axis, *TF* is the amount of flexure with the tube horizontal, and ζ_{tel} is the zenith distance of the yoke, within which the tube is presumed to be drooping under its own weight.

The real law may be rather different. On the AAT the departure is so pronounced that a $\tan \zeta$ model works better than $\sin \zeta$ (this is *not* due to any shortcomings in the refraction correction). Such a model was used for a time but has since been replaced with a (possibly more mechanically realistic) combination of the basic $\sin \zeta$ law together with an empirical $\tan^2 \zeta$ term. The empirical term, which is important at large zenith distances, was determined from the combined data of many pointing tests.

Like the refraction model, the above equation is the wrong way round for our purposes: we want to compute ζ_{tel} from ζ. However, the size of the effect (my guess is that *TF* will be no more than 10 arcsec) and its empirical nature make it unnecessary to do other than an approximate inversion. Thus the proposed model is:

$$\zeta_{tel} \approx \zeta - TF \sin \zeta$$

The minus sign has been chosen to be consistent with past practice (e.g. at the AAT). A droop would produce a positive value for the coefficient *TF* in the above formulation.

In principle, any vertical correction should be rotated into the mount frame by taking into account the tilt of the azimuth axis. However, where the azimuth tilt produces a large rotation – at the zenith – the vector is small, and unless both the flexure and the tilt are enormous, the rotation will produce a negligible effect and may be ignored. This convenient approximation obviously cannot be made in the case of an equatorial mounting.

Correction for tube flexure can be applied by means of the following Fortran algorithm. The vector `X1,Y1,Z1` is this time the post-collimation $[A, E]$, and the result, `X2,Y2,Z2`, is the mount $[A, E]$.

```
F = 1D0-TF*Z1
X2 = X1*F
Y2 = Y1*F
Z2 = (Z1+TF)*F
```

If the functional form of the tube flexure is not of the assumed form the required code may of course be very different.

Despite the advantages of working in Cartesian coordinates, the final phase of the collimation correction, and the whole of the tube flexure correction, may be better done in terms of spherical coordinates. Such an approach will be essential if the telescope tube is expected to pass through the zenith during service.

5.9 Encoding Errors

Little can be said in advance about the character or likely size of the corrections which will have to be applied to the encoder readings before they can be used by the servo software to close the position loop.

For each of the two sets of encoders there will, of course, be a zero point, and I will use the names *IA* and *IE* (azimuth and elevation index errors). As mentioned in the section on collimation errors, there will be two elevation zero point errors in the system (*CE* and *IE*) which cannot be separated by pointing tests. Thus there must be arrangements for resetting the encoders to a standard relationship with the machinery (possibly so that *IE* can be assumed zero and never has to be determined even after encoder replacement or other major disturbance). Although the azimuth index error will, in contrast, be available from pointing fits it would also be tidy to have a mechanical method of setting that encoder as well, to keep *IA* small and the encoder consistent with the gross telescope. Stability of the *IE* value will be crucial if large and rapidly changing encoder or gear errors are discovered.

The main gear errors will probably be accurately described by harmonics of one revolution of the axis and of each of the pinions or rollers, each harmonic requiring a cosine and a sine term.

Further smooth irregularities may be treated simply as higher harmonics; harsher techniques – empirical functions, lookup tables, etc – may be needed to cope with localized blemishes and with encoder errors.

Pointing tests can be expected to determine well the low frequency terms, but pinion errors and localized effects may need specially designed tracking tests instead (perhaps using an autoguider). Though possibly hard to measure, the errors ought to be very stable.

(I recommend that deliberate small random variations are introduced when setting the telescope to the standard park positions. This was not done on the AAT and there is now a 1 arcsec glitch in HA tracking through the meridian, thought to be because the gears have worn along the line of contact corresponding to zenith park as a result of longitudinal motion during oilpad pressure changes.)

6 INSTRUMENT MOUNT POSITION ANGLE

Commands and other controls will be required to allow the user to specify the orientation of the instrument mount. I suggest the following functions:

- In the tracking reference frame, set the rotator y-axis to a given angle relative to the meridian which passes through the rotator center. 'Meridian' means Northwards for equatorial coordinates, up for $[A, E]$. There should be 'go and stop' and 'go and track' options.
- In the observed $[A, E]$ frame (irrespective of the tracking frame), set the rotator y-axis to a given angle relative to the vertical which passes upwards through the rotator center. There should be 'go and stop' and 'go and track' options.

Either of the set functions should be available via direct pushbutton control of the rotator (so there would be a 'let me set the rotator by pressing buttons and then track the vertical' function for example).

Standard formulae giving the parallactic angle for the current $[h, \delta]$ of the target allow a good first order estimate of the required position angle. However, the accuracy goals mean that refraction and at least some of the telescope pointing corrections need to be taken into account.

Refraction has a substantial and variable effect on the geometry of the field as the latter is tracked across the sky, producing a vertical compression of the picture, the amount and orientation in equatorial coordinates of which vary as the track proceeds. This distortion cannot of course be removed by controlling the rotator angle, but its effect in terms of star trails will be reduced to an acceptable level if on the rotator the North-South line (in the tracking reference frame) as affected by refraction is kept to a constant orientation.

For an equatorial mount, the telescope pointing effects produce, to first order, a fixed offset in field orientation that is relatively innocuous. This advantage is not enjoyed by the $[A, E]$ design, where the size of the pointing corrections, and their orientation relative to equatorial coordinates, change during tracking. The largest effects on the position angle of the telescope field will be at the zenith, where the collimation corrections may swing the azimuth many degrees from the nominal value, and the position angle with it.

¿From the form of the refraction and telescope pointing corrections it would be possible to devise analytic expressions which would allow compensation of the rotator position angle demands. However, if the telescope tracking calculations are carried out as a series of coordinate transformations in x, y, z, a corrected position angle can be determined from the net x, y, z transformation without resorting to too much trigonometry. The steps are as follows:

1. Generate an 'up vector', a unit vector normal to the target vector in the tracking coordinate system and in the direction of the positive pole.
2. Transform it through the telescope pointing flow, up to but not including the collimation terms, to express it in the pre-collimation mount $[A, E]$ frame.
3. Express the direction of the y-axis of the rotator for mechanical position angle zero as a unit vector in the pre-collimation mount $[A, E]$ frame. Call this the rotator zero vector.
4. Determine the angle between the transformed up vector and the rotator zero vector.
5. Add corrections for collimation.
6. Combine with the desired orientation of the y-axis relative to the up vector to give the required instrument rotator position angle.

The algorithm in Fortran, starting from the target vector X,Y,Z, is as follows:

```
*  Generate the up vector
      RU = SQRT(MAX(X*X+Y*Y,1D-10)
      SB = Z/RU
      XU = -X*SB
      YU = -Y*SB
      ZU = RU
```

This up vector and the target vector are now processed through the appropriate part of the pointing flow giving transformed up and target vectors XUT,YUT,ZUT and XT,YT,ZT. Then:

```
*  Generate the rotator zero vector
      RT = SQRT(MAX(XT*XT+YT*YT,1D-10)
      SBT = ZT/RT
      XR = -XT*SBT
      YR = -YT*SBT
      ZR = RT

*  Angle between the up vector and the rotator zero vector
      SQ = XT*YR*ZUT+YT*ZR*XUT+ZT*XR*YUT
     :    -ZT*YR*XUT-YT*XR*ZUT-XT*ZR*YUT
      CQ = XR*XUT+YR*YUT+ZR*ZUT
      IF (SQ.EQ.0D0D.AND.CQ.EQ.0D0) CQ=1D0
      Q = ATAN2(SQ,CQ)
```

SQ and CQ, which stand for $\sin$ Q and $\cos$ Q, are respectively the scalar triple product of the T, R and UT vectors and the scalar product of the R and UT vectors. The sign of Q has been chosen so that for the full pointing transformation Q is to first order equal to the parallactic angle.

Allowance must now be made for the effects on the orientation of the rotator caused by (i) the mount movement arising from collimation corrections and (ii) the az/el non-perpendicularity. Using the nomenclature of the earlier section on collimation, the correction is approximately:

```
      QC = Q-Z2*ATAN2(XI,SQRT(RXY2-XI2)+NPAE*RXY
```

For a desired orientation on the sky of the y-axis of the rotator, PA (reckoned North through East), the demand rotator angle is PA minus QC.

Many of the quantities required in the above algorithm are also needed for the pointing algorithm itself, and so it is most convenient if the two sets of calculations are carried out together.

Both the pole case and the zenith case require special treatment to avoid arithmetic problems, a manifestation of the fact that here the position angle change is indeterminate. There are other practical considerations in both these cases.

Very near the zenith, for large collimation values, there may be difficulties, because the telescope pointing transformation depends on the actual rotator position angle, which, if the rotator is tracking, will depend on the telescope pointing transformation. However, the system ought to be stable except for a small area near the zenith well within the region where rapid azimuth motion makes observing impossible anyway.

The celestial pole also poses some problems as in this case it is not meaningful to talk of 'North'. The solution when observing the pole with a 2D detector (for example when making a photographic exposure) is to specify a pointing axis some distance off-center and to make the target position a point on the sky about that distance from the pole and with a Right Ascension chosen to determine the orientation of the field on the detector.

7 PRACTICAL DETAILS

7.1 Pointing Adjustments

The horizontal and vertical collimation *CA* and *CE* are of special importance as they cope with a wide range of physical effects which might otherwise pose an overwhelming calibration problem. Examples include tilts in the primary, secondary and tertiary optics, the particular positions of the instrument mounts at different foci, the effect of the atmospheric dispersion compensators, and thermal distortions of the telescope tube.

For this reason, I propose that these two numbers, *CA* and *CE*, are the ones adjusted when the telescope pointing is checked at the start of observing.

Utility routines will be required to set and calibrate *CA*,*CE*. On the AAT system they are called COLLIM and SNAFU respectively. COLLIM just reports the current values and allows new ones to be entered. During each run SNAFU keeps a record of the stars done so far and reports the mean and RMS for the two corrections. The name of the SNAFU program reflects the essentially empirical nature of the operation – in principle the AAT and Keck Telescope systems are absolute and should work from cold without calibration. During the running of the Keck equivalent of SNAFU, the instrument rotator should be kept stationary with respect to the telescope tube, unless the pointing axis x, y relative to the rotator axis is accurately known. If a Nasmyth focus is in use NRX,NRY must be known as well as the pointing axis x,y.

7.2 Calibrating the Pointing Axis Positions

The online measurement and adjustment of the position XIM,YIM of each pointing axis will be a key part of the calibration process and crucial if the pointing RMS delivered by pointing analyses is to be available to telescope users. The correct procedures must be followed even though various 'quick and dirty' techniques might initially appear to be easier (setting the *CA* and *CE* values to correct the pointing locally for example).

Provision for several pointing axes will be useful, and more if the chopping secondary is used in conjunction with a multi-aperture instrument. Depending on the nature and speed of the telescope command language, the simplest approach might be to allow the user to define a separate command for each required axis, perhaps also available via a pushbutton, that simply asserts the new XIM,YIM; something like:

```
A = "AXIS -1.341 -0.020"         ! define beam A
B = "AXIS +0.993 -0.017"         ! define beam B
A                                ! select beam A
etc
```

On the AAT, three axes are allowed for as standard, driven from reserved pushbuttons, from software, from simple digital interfaces and from the interprocessor link. Two of the axes are called 'aperture A' and 'aperture B', and the third 'reference axis'. The reference axis is the position of a fixed calibration mark in the TV system (something that should preferably be built into the Keck system from the start) which is calibrated relative to the rotator axis by determining its XIM,YIM twice with a 180° rotation in between and averaging the result.

The calibration program polls a set of pushbuttons (for example a hand paddle: up/down/left/right plus speed control are required) and directly increments and decrements XIM and YIM. The operator first positions a star on a known and accurately calibrated axis position (probably the TV reference axis mentioned above), then selects the new axis (by pressing the 'aperture A' button, say) and then uses the buttons to 'guide' the star onto the instrument aperture. The tracking $[\alpha, \delta]$ remains fixed while the real telescope moves as a result of the changing collimation corrections; once the star is on the instrument aperture the calibration is complete. Once set up, the system will track a star with full accuracy anywhere in the focal plane, even if the rotator position angle is changed.

In the AAT software, pressing the buttons while the reference axis is selected has the effect of changing the telescope demand $[\alpha, \delta]$ while making equal and opposite changes in XIM,YIM for all the other axes. This is to allow tracking errors to be eliminated during the calibration run.

A refinement in the AAT 'APOFF' calibration program that has been found very useful for IR work is the 'MAXIM mode', where the beam is automatically centered on the source. A ± 10 volt ADC is provided to which the observers connect the output from their detector. The program samples the ADC while performing a cross scan, self-convolves each arm to give the point of best symmetry (which works even if the signal is negative-going), plots the profiles, and adjusts the pointing axis x, y so that the physical telescope moves to align the beam to the source. Each run takes about 30 sec, and gives more consistent results than manual peaking-up. (The same automatic centering is available during normal observing via the MAXIM program proper. This simply alters the telescope demand position, not the pointing calibration.)

7.3 Economical Implementation

There is much to be said for doing the full pointing calculation at or near servo rates; computing power has fallen in cost dramatically in recent years, and the software will be as compact and obvious as it is possible to achieve. However, there are counterarguments: computing power is still not free, the pointing software may actually be easier to follow if a lot of the complicated 'background' calculations are kept out of the main flow, and it will be possible later to increase the loop speed if this turns out to be desirable.

A simple interpolation scheme has already been suggested for computing the sidereal time. In the case of the pointing transformations, I propose a scheme which combines rigor and precision with economical use of CPU time by representing the bulk of the telescope pointing transformation as slowly changing 'osculating transformation matrices' (OTMs) which can be recalculated relatively infrequently and used as interpolation devices by the fast loop. An OTM can represent the net effect of several arbitrarily complicated and rigorous pointing models as long as the transformations are locally smooth (which is in any case a requirement if the pointing is ever to be accurate and stable).

The pointing calculations can then be done in three groups as follows.

At low frequency; about once every five minutes is more than enough:

- TDB-TT
- check incremental software sidereal clock
- Earth barycentric position and velocity
- precession/nutation matrix, aberration vector, etc
- refraction parameters
- thermal effects?

At medium frequency; limited by how far the telescope can offset in between iterations, and hence how far out the pointing predictions will be – about once every five seconds is satisfactory:

- Generate 1st OTM: mean $[\alpha, \delta]$ to apparent $[\alpha, \delta]$
- Generate 2nd OTM: apparent $[-h, \delta]$ to pre-collimation $[A, E]$

- amount of tube flexure if complicated function

At high frequency; say 20 Hz, to allow differential movements to exploit the full bandwidth likely to be available:

- tracking coordinates to apparent $[\alpha, \delta]$ (using 1st OTM)
- to apparent $[-h, \delta]$
- to pre-collimation $[A, E]$ (using 2nd OTM)
- collimation
- to mount $[A, E]$
- analogous computations to get rotator angle
- increment sidereal time

Two osculating transformation matrices are needed because the Earth rotation is clearly not a slowly changing effect. Similarly, the second OTM cannot include the collimation adjustments because the collimation model can change abruptly (when switching a star image from one instrument aperture to another for example, which is done by changing the pointing axis x, y parameters).

The encoder computers will have an analogous but hopefully much simpler transformation to perform, and it could well turn out to be best (for example) to compute the slowly varying corrections at a lower frequency than reading the encoders and issuing the corrected reading to the servo software. Use of matrices will not be required in this case as each system has only to deal with one coordinate.

An osculating transformation matrix is easily determined once the procedure for transforming a target vector through the relevant part of the pointing flow is available. The steps are as follows:

1. Generate three 'probe vectors' surrounding the target vector at a distance over which the distortions in the coordinate system do not depart seriously from linear scaling and shearing. The precise positions are unimportant but for good sampling of the transformation field should be reasonably evenly spread, and not so close that the numerical precision is significantly eroded. An equilateral triangle a few arcminutes a side works well, and a procedure for generating such a pattern is given later.
2. Transform the probe vectors, one by one, through the part of the pointing flow which is to be modeled.
3. Use the resulting three x-values to solve for three coefficients which enable each x-value to be expressed as a linear combination of the original x, y, z of that vector. Do the same for y and z.
4. The nine coefficients can be laid out as a 3×3 matrix by which any of the original probe vectors can be multiplied to yield the corresponding transformed probe vector.

If the pointing transformation were a pure rotation, the osculating transformation matrix would be orthogonal (except for rounding errors) and would correctly transform not just the probe vectors but any other vector anywhere in the sky. Where the transformation also includes an element of distortion (due to refraction for instance), the matrix will be nearly but not quite orthogonal, will analytically transform only the three probe vectors, but for a smoothly changing transformation will give a close approximation for any position in the neighborhood.

Expressed symbolically, for three probe vectors P_1, P_2, P_3, the pointing transformation will yield a further three vectors Q_1, Q_2, Q_3. For an osculating transformation matrix $a, b, c, \ldots$ as follows:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{Q_{1\rightarrow 3}} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_{P_{1\rightarrow 3}}$$

the matrix elements $a, b, c, \ldots$ can be determined from the following three sets of simultaneous equations:

$$\left.\begin{array}{lclclcl} x_{Q_1} & = & x_{P_1}a & + & y_{P_1}b & + & z_{P_1}c \\ x_{Q_2} & = & x_{P_2}a & + & y_{P_2}b & + & z_{P_2}c \\ x_{Q_3} & = & x_{P_3}a & + & y_{P_3}b & + & z_{P_3}c \end{array}\right\} \longrightarrow a, b, c$$

$$\left.\begin{array}{lclclcl} y_{Q_1} & = & x_{P_1}d & + & y_{P_1}e & + & z_{P_1}f \\ y_{Q_2} & = & x_{P_2}d & + & y_{P_2}e & + & z_{P_2}f \\ y_{Q_3} & = & x_{P_3}d & + & y_{P_3}e & + & z_{P_3}f \end{array}\right\} \longrightarrow d, e, f$$

$$\left.\begin{array}{lclclcl} z_{Q_1} & = & x_{P_1}g & + & y_{P_1}h & + & z_{P_1}i \\ z_{Q_2} & = & x_{P_2}g & + & y_{P_2}h & + & z_{P_2}i \\ z_{Q_3} & = & x_{P_3}g & + & y_{P_3}h & + & z_{P_3}i \end{array}\right\} \longrightarrow g, h, i$$

Note that the equations can be solved by inverting just one matrix and then multiplying by the three transformed probe vectors in turn. The matrix will be ill-conditioned: the cofactors and the determinant will all be the result of subtracting nearly equal quantities, because the three rows will be very similar. (It would be singular if any two of the probe vectors were coincident – obviously.) However, the degree of ill conditioning is such that with double precision arithmetic the result will be of more than adequate accuracy. Algorithms for solving such sets of linear equations are widely available; the Starlink SLALIB library has a matrix inversion routine DMAT which is suitable. Alternatively, a standard 3×3 matrix inversion algorithm can readily be hard-coded.

The Fortran algorithm which follows generates three suitable probe vectors [`X1,Y1,Z1`], [`X2,Y2,Z2`] and [`X3,Y3,Z3`] starting from a target vector [`X,Y,Z`] and a radial distance `DEL`. A `DEL` value of 0.005 radians (about 1000 arcsec) gives good results, as will a wide range of other values. The pattern is a fairly accurate equilateral triangle centered on the target, but this is similarly uncritical. It is, however, advisable to keep the vectors at unit length as a precaution against incompatibility with the pointing calculations. The method of doing this used below is approximate, but entirely adequate. At the expense of more CPU time each component of a probe vector could simply be divided by the modulus `SQRT(X*X+Y*Y+Z*Z)` of that vector.

```
*
*  Generate probe vectors
*

*  Useful functions
      R = SQRT(X*X+Y*Y)
      IF (R.GE.1D-10) THEN
         SA = Y/R
         CA = X/R
      ELSE
         SA = 0D0
         CA = 1D0
      END IF
      SASB = SA*Z
      CASB = CA*Z
```

```
*  X,Y,Z shifts for generating the three probe vectors
      DUP = DEL
      DDN = DEL*0.5D0
      DRL = DEL*0.8660D0

      DXUP = -DUP*CASB
      DYUP = -DUP*SASB
      DZUP = +DUP*R

      DXDN = -DDN*CASB
      DYDN = -DDN*SASB
      DZDN = +DDN*R

      DXRL = -DRL*SA
      DYRL = +DRL*CA

*  Normalization factor
      F = 1D0-DEL*DEL/2D0

*  First probe vector: above the target
      X1 = F*(X+DXUP)
      Y1 = F*(Y+DYUP)
      Z1 = F*(Z+DZUP)

*  Second probe vector: down and to the right
      X2 = F*(X-DXDN-DXRL)
      Y2 = F*(Y-DYDN-DYRL)
      Z2 = F*(Z-DZDN)

*  Third probe vector: down and to the left
      X3 = F*(X-DXDN+DXRL)
      Y3 = F*(Y-DYDN+DYRL)
      Z3 = F*(Z-DZDN)
```

7.4 Summary of Pointing Data Requirements

The computation of the demand azimuth, elevation, velocities, and position angle requires three sorts of information: external, measured, and user-entered.

The external data are as follows:

- UTC
- telescope longitude, latitude, and height
- polar motion x, y
- UT-UTC
- TT-UTC

The measured data include the following; there will be additional pointing coefficients, which cannot readily be identified until pointing tests are done.

- temperature, pressure, humidity
- AX: azimuth axis tilt, South
- AY: azimuth axis tilt, East
- $NPAE$: Az/El nonperpendicularity
- CA: horizontal collimation
- CE: vertical collimation zero
- focal length
- NRX,NRY: Nasmyth rotator displacement
- rotator position angle, actual
- pointing axis x, y (several)
- TF: tube flexure
- Gear error parameters
- Encoder error parameters
- IA: azimuth index error
- IE: elevation index error

The list of essential user inputs is as follows; there will also be offsets from base and non-sidereal track rates etc.

- target coordinate system (system and in some cases equinox)
- tracking coordinate system (likewise)
- target coordinates (perhaps including proper motions etc)
- pointing axis selection
- required orientation of rotator y-axis