Rutherford Appleton Laboratory
Particle Physics & Astronomy Research Council
Gemini Project
Gemini TCS Note (PTW) 3.12 (TN_PS_G0044)

TCS/PTW/3.12 (TN_PS_G0044)

P. T. Wallace

23rd December 1996

Gemini Pointing Algorithms

Contents

1 INTRODUCTION

This paper presents algorithms for open-loop pointing and tracking of the Gemini 8-metre altazimuth telescopes.[1] We discuss the relevant standard timescales and reference frames, and give recipes for achieving the required transformations. Various telescope pointing corrections are also taken into account; however, a detailed description of the pointing models is not yet available and in any case has little impact on the overall computational strategy, which is the subject of the present paper.

Vector and matrix methods are used throughout. Despite an emphasis on rigour, efficiency considerations are not overlooked, and the strategy includes techniques for avoiding unnecessary consumption of computer time.

To avoid spelling out well-established (and tedious) positional-astronomy procedures, the present report assumes the availability of the Starlink SLALIB library (see Starlink User Note 67). The SLALIB routines are available from Starlink in Fortran or from the author in a proprietary C version that is available free to non-profit research institutes (including the Gemini project). The Fortran version is ANSI-standard apart from minor and commonplace infringements (mostly covered by the US Department of Defense extensions). The C version is ANSI-standard throughout. Other relevant Starlink software includes the COCO star coordinate conversion utility (SUN/56) and the TPOINT telescope pointing analysis package (SUN/100). The COCO program contains a prescription for most of the star coordinate transformations that will be needed to control the Gemini telescopes and may be useful for reference; TPOINT is a possible tool for analyzing the pointing of the Gemini telescopes during commissioning and operation, or its algorithms could be used in the construction of new tools and techniques. package (SUN/100). Both COCO and TPOINT are available in Fortran. In the case of COCO, there is no C version; a proprietary C version of TPOINT exists but there are, at present, no arrangements for Gemini to use it.

Key parts of the algorithm are illustrated with sample code. Fortran is used, though the operational implementation uses C (plus EPICS) exclusively. More complete sample code, this time written in C, is set out in the document TCS/PTW/4, together with worked-out numerical examples.

[1]The methods used are similar to those in *Proposals for Keck Telescope Pointing Algorithms* by P. T. Wallace.

n.b. This document was first published before the Gemini TCS code was developed; the latter is always referred to in the future tense here, even though later revisions of the present report appeared after the TCS software was developed.

2 GENERAL STRATEGY

2.1 Basic Transformation Flow

Figure 1 identifies the main steps the TCS will use to transform a position in the user's chosen coordinate system (called 'tracking coordinates') into one that the MCS 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, and to what extent interpolation techniques will be used to keep CPU-time consumption at reasonable levels; we will return to these topics later.

In Figure 1, the coordinates marked $\rightarrow$ are those appropriate for applications software to supply as demands, including scan patterns and adjustment *via* the TCS handset, 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 TCS-to-MCS 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. The frequency at which the MCS causes the servo systems to compare demanded position with actual position will be considerably higher, involving interpolation of the demands from the TCS. Further details of this process are outside the scope of the present paper.

The philosophy presented here is to carry out all the positional astronomy calculations rigorously, 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 will be relaxed.

2.2 The Virtual Telescope

The tracking coordinates are an interface to the *virtual telescope*, a simulation of an ideal device produced by hiding, as far as it makes sense to, the defects of the real telescope under layers of software. The virtual telescope is generally the only one that should interest the astronomer and which data-acquisition procedures should deal with, but there will be some respects in which display or control of the *real* telescope is required. For example:

1. 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

→	MEAN $[\alpha, \delta]$	↓
	↓	↓
	J2000 FK5 MEAN $[\alpha, \delta]$	↓
	↓	↓
	annual aberration	↓
	light deflection	↓
	precession/nutation	↓
	↓	↓
→	APPARENT $[\alpha, \delta]$	↓
	↓	↓
	Earth rotation	↓
	↓	↓
	APPARENT $[-h, \delta]$	↓
	↓	↓
	diurnal aberration	↓
	↓	↓
	TOPOCENTRIC $[-h, \delta]$	TCS
	↓	20 Hz
	$[-h, \delta]$ to $[Az, El]$	↓
	↓	↓
→	TOPOCENTRIC $[Az, El]$	↓
	↓	↓
	refraction	↓
	↓	↓
	OBSERVED $[Az, El]$	↓
	↓	↓
	azimuth axis tilt	↓
	↓	↓
	Az/El nonperpendicularity	↓
	position of instrument rotator	↓
	pointing-origin	↓
	tube flexure	↓
	↓	↓
→	MOUNT $[Az, El]$	↓
	↓	↓
	AZ/EL SERVO SYSTEMS	
	↑	↑
	CORRECTED ENCODER $[Az, El]$	↑
	↑	↑
	roller/gear/centring errors	$\gg$ 20 Hz
	encoder index errors	MCS
	encoder errors	↑
	↑	↑
	RAW ENCODER $[Az, El]$	↑

Figure 1: **The TCS/MCS Pointing Flow**

The set of transformations shown describes the relationship between the target position supplied to the TCS (one of those marked →) and the desired telescope encoder readings received by the MCS. There are two major transformations: $[\alpha, \delta]$ to $[-h, \delta]$, and $[-h, \delta]$ to $[Az, El]$. The others are all minor.

at a readout of mount [Az, El], but will expect a readout of distance or time to go, and perhaps a picture of the telescope orientation. The TCS could produce an 'in position and tracking' status once both the position and velocity errors fall within specified limits. The status is accessible to data-acquisition systems (which as well as knowing when to start an exposure might be able to close the shutter during wind gusts or moments of poor seeing) and will also be displayed on the TCS control console to inform the telescope operator.

2. Cable wraps and mechanical interference phenomena will affect slewing strategy, and the TCS will require advice from outside about when to go the long way round. This might come from the OCS, based upon future targets for example, or through the TCS control console.

3. Users will expect and need real-time displays of mount dynamics and guiding activity, and the virtual-telescope philosophy in no way discourages such information being available (though it may be subsystems such as the MCS and A & G that actually provide such displays, rather than the TCS).

2.3 Base and Offset from Base

Functions which move the telescope will be able to specify the tracking position (one of those 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 demanding software could equally well do, the value of having the TCS provide an offset-from-base mechanism as part of the tracking system relieves the demanding software of remembering where it started (useful when cleaning up after an abort) and also helps the telescope operator understand what is going on during a complex scanning or offsetting manoeuvre.

A base and offset concept is also valuable when specifying the pointing-origin positions 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, which will be used as a check during TCS software development. It is probably worth doing full COCO-style transformations even though this may appear to be excessively fussy. Certainly offsetting from nearby bright stars will be more assured if small effects such as parallax have been thought about and 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. Thus the $[\alpha, \delta]$ which starts off the flow is the position at the current epoch in the nominated tracking reference frame.

Observers will want to work in several different reference frames, in some cases, it has to be said, 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 $300''$ due north to a particular spot in the Nebula rightly expects α to stay fixed and δ to change by $300''$; 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 $1''$ from the correct point, due to the 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, compared with past practice.

Obvious tracking reference frames to consider include not only equatorial and altazimuth coordinates but also ecliptic, galactic, *etc.* However, there is considerable doubt whether the latter options are really useful, and they have been left out of the initial Gemini design. (They may, however, be useful for information displays and logging.) The required transformations are all in COCO's repertoire and there will be no problem in providing them if users really want them. It is possible some radio-astronomers might want an exotic flavour 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, the Gemini TCS will 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.
- (Topocentric) apparent $[\alpha, \delta]$, new style. This would be a suitable form for the TCS to accept input from planetary ephemeris programs, which would have already allowed for parallax (annual and diurnal) 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).
- Observed $[Az, El]$. This may be useful occasionally where a data-acquisition application wants to do all the other transformations itself – satellite tracking would be an example.
- Mount $[Az, El]$. The obvious applications are engineering ones – parking the telescope for example.

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

When entering target data, there will be both separate commands for specifying the coordinate system and for entering coordinates, or the target coordinate system will be specified by parameters supplied along with the coordinates.

3.2 Some Remarks on Target Mean Place Data Entry

The way input coordinates are supplied by the user depends heavily on the details of the TCS control console displays and the interfaces to the OCS, the details of which have yet to be established. However, some general comments are possible 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 we will call FK4 and FK5 for short, will need to be supported. Very often the telescope user 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* (short for 'epoch of mean equator and equinox').
- The *epoch* (time zero for working out the proper motion correction, and not to be confused with the equinox).
- Proper motion (various formats are required).
- Parallax (arcsec).
- Radial velocity (km/s).

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

Depending on the details of TCS control console displays *etc*, defaulting conventions along the following lines will be employed:

- 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 will 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 will also be accepted as year, month, day. If a year, for formality's sake a B or J prefix can be used as for the equinox, though this will have a negligible effect on the result. The value will default to that of the equinox, which is almost always the case in star catalogues. The epoch will *not* be allowed 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 we set out 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 $[Az, El]$, 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, when a new target is requested, 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 acquiring the target, 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 will obviously be scope for optimization and approximation in the above procedures, if this proves 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, in the first instance the transformations will be 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 when a new target is requested (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 ($1\rlap{.}''74$) it falls off rapidly and has shrunk to about $0\rlap{.}''02$ at an elongation of 20° from the Sun, which is presumably closer than will ever be used on Gemini (unless a total Solar eclipse is in progress). The effect is thus negligible for our purpose and could be omitted. However, unless CPU time is at a premium, it will be best to perform the correction for the usual reasons of (a) rigour and (b) to assist comparison with other software.

The annual aberration is a function of the Earth's velocity relative to the solar system barycentre (available through the Starlink SLALIB routine EVP) and produces shifts of up to about $20''\!.5$.

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 could 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 will be to precompute the target-independent parameters with the MAPPA routine and then to use MAPQKZ.

4 TIME AND POLAR MOTION

The current time expressed in the following three systems is 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 observers 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 that will be available through the TCS control console).
- TDB (barycentric dynamical time) is needed for various dynamical calculations (*e.g.* planetary predictions) and will be available to data acquisition applications for timing variable sources. (TDB is one of family of timescales, along with terrestrial time TT and the obsolete ephemeris time ET, which for most Gemini purposes are the same thing but which differ from UTC by of order 1 minute at present.)

These timescales are, in principle, quite unconnected, and one cannot rigorously be converted into another without additional information. ST is a function not of UTC but of UT1, and the difference $\Delta\mathrm{UT} = \mathrm{UT1} - \mathrm{UTC}$ has to be obtained from data published by the International Earth Rotation Service (details below). UTC contains "leap seconds", and cannot be used directly in astronomical formulae. If the current offset from the continuous timescale TAI (International Atomic Time) is known, this in practice allows TDB to be computed, although TDB and TAI are not formally linked. In fact TAI is a more satisfactory choice than UTC for the fundamental time service of an observatory, because of the need to continue operations through a leap second, and we will assume this starting-point. Summarizing, the following are needed:

1. TAI.
2. $\Delta\mathrm{AT} = \mathrm{TAI} - \mathrm{UTC}$. This is an integer number of seconds, changing by 1 on each occasion that a leap second occurs.
3. $\Delta\mathrm{UT} = \mathrm{UT1} - \mathrm{UTC}$. This is a fraction of a second, changing rather unpredictably by a few milliseconds a day.

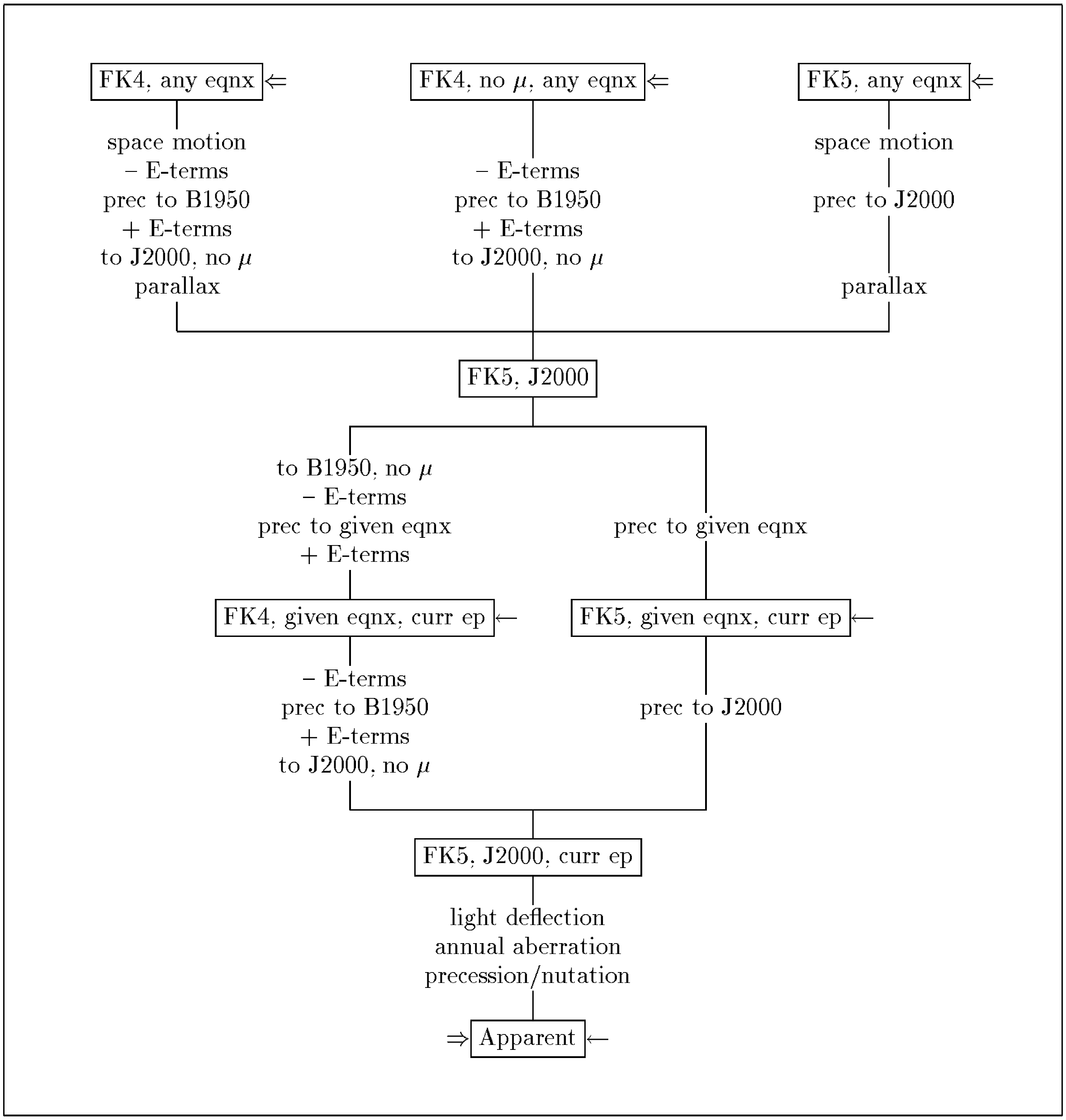

Figure 2: Transformations for mean $[\alpha, \delta]$

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 need to be executed only once per new target (in practice), but all the transformations from that level down have to be performed at the full pointing rate.

4. The geographical position of the Earth's axis of rotation.

5. The telescope mean longitude and latitude.

Item 1 is provided by the observatory time service. Items 2–4 are published in International Earth Rotation Service (IERS) *Bulletin A*. See `http://maia.usno.navy.mil` on WorldWideWeb for further details. Item 5 comes from conventional surveying.

The correction of the telescope's longitude and latitude for polar motion is barely significant (rarely above $0\rlap{.}''3$, though it does move around on timescales of a few months) and need be made only at the start of the observing session (if at all). The computations will require the polar x, y, the telescope mean longitude, and the telescope mean geodetic latitude. The simple first order expressions in section 3.27-3 in *Explanatory Supplement to the Astronomical Almanac*, P. K. Seidelmann (ed), 1992, (hereafter referred to as the 'ES') are adequate.

The accuracy requirement for the geographical position of the telescope isn't particularly high, so that the figures provided by the surveyors will be more than adequate. If an error of a few arcseconds did occur, it would be automatically corrected as part of the azimuth axis misalignment measured through pointing tests. A gross error would produce an apparent asymmetry in the atmospheric refraction, and even this would tend to be absorbed into other parts of the pointing model. The reason that it is nonetheless worth considering correcting for polar motion, despite the small size of the effect, is that this will help when models determined years apart are being compared; an unknown but fixed error in the telescope position would not affect such comparisons.

There is a subtle point here to do with azimuth, namely where 'north' is. If the conventional view is taken that polar motion causes variations in the telescope's latitude and longitude only, the resulting predictions of telescope azimuth will naturally refer to the celestial ephemeris pole, rather than the terrestrial pole, the two being in relative motion as we have seen. However, because the telescopes are fixed to the ground, their azimuth zero-point correction – relative to celestial north – would then be subject to change due to polar motion. This effect, of the same sort of size as the polar motion itself, can be allowed for in the pointing calculations, and this is the approach we will take. It should be borne in mind, however, that the azimuth polar-motion correction may not have much practical significance given the *ad hoc* adjustments to azimuth zero-point that traditionally occur as part of nightly calibrations using stars. But if it turns out that the Gemini azimuth "zero-sets" are ultra-stable, then allowing for the effects of polar motion in this way will be beneficial; under these circumstances, it will be best to exclude azimuth zero-point from the nightly calibration procedures, leaving left-right collimation error to mop up any observed sideways bias caused, for example, by thermal distortions in the telescope structure.

The ΔUT tables given in IERS Bulletin A will need to be interpolated to give a revised value for each day. A simple linear interpolation will be good enough for pointing the Gemini telescopes. Should the supply of new tables be interrupted for any reason, extrapolation may be necessary. Depending upon how long into the future the predictions have to be made, linear extrapolation may not be good enough; more sophisticated and accurate extrapolations exist which take into account known seasonal effects.

As new Internet-based information services are introduced, the automatic retrieval of ΔUT once a day may become feasible.

TDB can be determined as follows:

1. To the TAI add $32^{\mathrm{s}}.184$ giving TT.
2. Using the Starlink SLALIB routine RCC, add the small corrections for gravitational red-shift and transverse Doppler effect (peak to peak $0^{\mathrm{s}}.0033$, clearly of no significance for telescope pointing but probably worth doing partly for rigour but mainly as a service to data acquisition applications involving timing of pulsars).

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 UTC by subtracting ΔAT from TAI.
2. Compute UT1 by adding ΔUT to the UTC.
3. Compute the Greenwich mean sidereal time GMST from ES equation 2.24-1. This is implemented in the Starlink SLALIB routine GMST (also GMSTA).
4. 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.
5. Subtract the telescope west longitude to give ST, the local apparent sidereal time.

It will not be necessary to go through the whole of the above calculation continuously as part of the telescope tracking. For example, the sidereal time may be implemented within the telescope control computer as a linear extrapolation working from a value computed for a nearby reference epoch. The sample ST and the corresponding reference epoch can be refreshed every few minutes.

Leap seconds, which happen around 0 hours UT on January 1 or July 1, are announced several months in advance in time bulletins. The following actions need to be taken:

1. When the announcement is first made, the TCS initialization file, or some other file that is accessible to the TCS, must be updated to show the date at which the leap second will occur. It can be assumed that the new ΔAT after that point will be one more than the current value being used by the TCS.
2. After the leap second – the next day is soon enough, and the system will continue to function satisfactorily for some days after that as long as ΔUT is left alone – these three changes will need to be made, all at once:
   - The date for the next leap second should be set to some arbitrary point in the distant future.
   - The current ΔAT must be incremented by 1^{s}.

- The current ΔUT must also be incremented by 1^s. (The IERS tabulations will already contain this discontinuity.) For example, if ΔUT was, say, $-0\overset{s}{.}853$ before the leap second, its value after the leap second will be $+0\overset{s}{.}147$. (The purpose of leap seconds is, of course, to keep ΔUT within about $\pm 1^s$, so that UTC can be used directly for astronavigation, getting up in the morning, and other relatively low precision positional-astronomy applications.)

5 POINTING TERMS

5.1 Vector Methods

All the telescope pointing calculations will be specified 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 rigour, maintain more uniform accuracy over the celestial sphere, and allow more succinct expression.

The coordinate convention we 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 $[\,Az, El\,]$. To avoid left-handed systems, we shall be using $[-h, \delta\,]$ internally, and Az 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 $A = +\pi/2, B = 0$. All external interfaces will use the normal conventions, with azimuth, for example, running from north through east.

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 an 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, which the Gemini telescopes cannot reach.

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 ($24^{\mathrm{h}} = 2\pi$ 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 will be up to about $0\overset{''}{.}3$ (for targets on the meridian, at Mauna Kea), 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 DIURAB (slightly different for the two Gemini sites), 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 $[-h, \delta]$ 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, followed by a small rotation about the z-axis to correct for polar motion. The main rotation is through 90° minus the astronomical latitude, corrected for polar motion. The second rotation, always very small, is equal to the required azimuth change. (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 centred 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:

$$\begin{bmatrix} +\cos\epsilon\sin\phi & \sin\epsilon & -\cos\epsilon\cos\phi \\ -\sin\epsilon\sin\phi & \cos\epsilon & +\sin\epsilon\cos\phi \\ +\cos\phi & 0 & +\sin\phi \end{bmatrix}$$

The symbol ϕ represents the telescope true geodetic latitude, and the symbol ϵ is the correction to azimuth due to polar motion: both can be computed by the Starlink SLALIB routine POLMO. Making the entirely acceptable approximations $\sin\epsilon \simeq \epsilon$ and $\cos\epsilon \simeq 1$, given that ϵ will never exceed a fraction of an arcsecond, we can write down the multiplications explicitly as follows (in Fortran):

```
X2 = +SP*X1+E*Y1-CP*Z1
Y2 = -E*SP*X1+Y1+E*CP*Z1
Z2 = +CP*X1+SP*Z1
```

`SP` and `CP` are $\sin\phi$ and $\cos\phi$ respectively; `E` is ϵ. `X1,Y1,Z1` is the topocentric $[-h, \delta\,]$ vector and `X2,Y2,Z2` the topocentric $[\,Az, El\,]$ vector.

Note that the south-through-east azimuth convention used here is different from the north-through-east scheme that is generally used and has been adopted for Gemini. This is to avoid introducing a left-handed coordinate system and hence sign changes in the procedures which convert between spherical and Cartesian coordinates. Transformation to the convention used on the Gemini telescopes (simply a sign reversal in x) will 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 modelled 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 colour of the source/detector combination.

For typical observing conditions at the Mauna Kea Gemini telescope (4253 metres above sea level), A will be approximately $+36''$ and B approximately $-0''\!.04$. The corresponding values at Cerro Pachón (2737 metres) are $+44''$ and $-0''\!.05$ respectively.

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 integration through a model atmosphere. The Starlink SLALIB routine REFCO computes A and B by calling, for two sample zenith distances, a routine REFRO which implements the algorithm given in section 3.281 of the ES. 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 start-up and every few minutes thereafter, by the Starlink SLALIB routine REFCO. The strategy for supplying the TCS with meteorological parameters has yet to be determined. Depending on the reliability of the measuring instruments, continuous updating from the enclosure control system may run the risk of introducing glitches in the tracking. It may be better to refresh the values only on request from the operator, or perhaps during telescope slews.

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 of borderline significance; for example at the minimum Gemini elevation of 15° the error is about $0\overset{''}{.}15$ (at the Cerro Pachón site). It is, however, possible to write out one iteration of the Newton-Raphson method to give an essentially perfect result (well under $0\overset{''}{.}001$ even in the worst case) at little extra computational cost:

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

When the vector formulae for refraction are set down, considerable simplification is possible, a by-product of which is the elimination of any problem at the zenith. The following Fortran procedure takes an *in vacuo* $[\,Az, El\,]$ vector `X1,Y1,Z1` and calculates the refracted position `X2,Y2,Z2`, the new coordinate system being observed $[\,Az, El\,]$. 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)
```

`D` is the change in elevation. The operational implementation may involve extra code to protect against divide by zero at the horizon: even though the real telescope itself cannot reach low elevations, commanding the virtual telescope to acquire such a target may involve executing the above code, leading finally to a 'star has yet to rise' status. A more elaborate version of the above algorithm is implemented in the Starlink SLALIB routine REFV.

In general, more than one set of refraction constants A and B will be needed, because of possible colour differences between science target and guide-stars and between the science detector and the wavefront sensors. Multiple sets of refraction constants are most efficiently generated by calling the Starlink SLALIB routine ATMDSP.

5.6 Tilt of the Azimuth Axis

The surveying techniques used to set up the azimuth bearing of each Gemini telescope 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 AN radians north and AW radians west 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:

$$\begin{bmatrix} +\cos AN & 0 & +\sin AN \\ -\sin AN \sin AW & +\cos AW & +\cos AN \sin AW \\ -\sin AN \cos AW & -\sin AW & +\cos AN \cos AW \end{bmatrix}$$

Using `SN,CN,SW,CW` to signify $\sin AN$, $\cos AN$, $\sin AW$, $\cos AW$, this can be coded in Fortran as follows:

```
X2 =  CN*X1+SN*Z1
Y2 = -SN*SW*X1+CW*Y1+CN*SW*Z1
Z2 = -SN*CW*X1-SW*Y1+CN*CW*Z1
```

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

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

(It may be useful at this point to note that the algorithm for an equatorial mounting would require a large rotation, from $[\,Az, El\,]$ to pre-collimation mount $[-h, \delta]$. The above matrix is rigorous and could be used for an equatorial. However, the AN value, instead of being small, would be close to $\phi - \pi/2$, 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-origin*

5.7.1 Az/El Nonperpendicularity

The $[\,Az, El\,]$ 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 $5''$, 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 we 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 it is almost impossible to separate the two from pointing tests. Incorrectly interpreting an elevation encoder index error as vertical collimation error would cause the prediction of projection geometry to be wrong; it will be best to assume that, following optical alignment, the CE value is zero for the rotator axis and to use IE to correct the pointing. Furthermore, although not essential, it will be best to arrange, through mechanical adjustment, that the corrected $[\,Az, El\,]$ encoder readings are reasonably close to the mechanical reality, so that IE is small.

CA for the rotator axis might be a few tens of arcseconds in size. Predictable and perhaps large corrections to CA and CE will be required for the shifts produced by an atmospheric dispersion compensator, if used.

5.7.3 Pointing-Origin

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

By 'pointing-origin' we mean the nominated point in the focal plane to which the pointing refers. The terms 'instrument aperture', 'beam', 'pointing axis' and 'optical axis' are sometimes used to mean the same thing. See the section on calibrating the pointing-origin positions, later.

The sign convention for RPA, XIM and YIM is as follows (see Memorandum to Gemini Project Managers from Earl Pearson dated 5th January 1994):

RPA is zero when, for elevations well away from the zenith, the projection on the sky of the rotator's y-axis points downwards (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-origin is below the projection of the rotator axis.

The positive direction of XIM is such that the XIM and YIM axes have the conventional $[\,x, y\,]$ orientation as seen looking down into the A & G from M2: therefore projected on the sky the XIM axis is, contrary to the usual convention, 90° anticlockwise of the YIM axis, so that for zero RPA and elevations well away from the zenith XIM is positive when the projection on the sky of the pointing-origin is to the right of the projection of the rotator axis.

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

XIM and *YIM* will be in units of length (metres is probably the formally correct unit, but millimetres is more convenient) 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 `FL` 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/FL
YR = YIM/FL
```

For pointing origins lying well off-axis (for example a WFS at the edge of the field), some adjustment to `XR` and `YR` will be essential in order to correct for the optical projection geometry, relative to the assumed tangent plane or *gnomonic* projection. The form of the adjustment will be simplest if the centre of the projection geometry (and hence the point of best image quality) is on the rotator axis.

5.7.4 The Combined Collimation Correction

All of the collimation effects just described 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 $0''.1$ – to yield a net pointing-origin position. In Fortran, the algorithm is as follows, with `X1,Y1,Z1` the pre-collimation mount $[\,Az, El\,]$, and where `XI` and `ETA` are the altazimuth counterparts of the 'standard coordinates' $[\,\xi, \eta]$ of spherical astronomy:

```
*  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-origin position (on sky XI +ve left, ETA +ve up)
      ETA = XR*SPA-YR*CPA+CE
      XI = -XR*CPA-YR*SPA+CA+NPAE*(Z1-ETA*RXY+TF*RXY2)
```

The expression '`Z1-ETA*RXY+TF*RXY2`', used in allowing for the non-perpendicularity, is a preliminary estimate of the mount elevation. For small values of `NPAE`, the approximate corrections for vertical collimation and for 'tube flexure' (the '`-ETA*RXY+TF*RXY2`' terms) can safely be omitted, and the simple expression '`NPAE*Z1`' used instead. The principal justification for these tiny corrections is to improve consistency between the 'downstream' (where to point the telescope?) and 'upstream' (where is the telescope pointed?) transformations.[2]

[2] A compromise between rigour and efficiency is to retain the '`-ETA*RXY`' term in the downstream transformation but to move the '`TF*RXY2`' term, which is always tiny, out of the downstream algorithm into the upstream algorithm, with the requisite change of sign. The errors introduced by this adjustment are of no practical significance, and the agreement between the downstream and upstream calculations is preserved. This approach is used in the Gemini operational code.

This overall pointing-origin position XI,ETA can now be used to derive a rigorous transformation predicting the required mount [Az, El] to align the pointing-origin 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 (provisionally assuming no tube flexure)
      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
```

In the 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-origin, or (ii) the net horizontal collimation. Case (i) would have to be handled properly if the telescope tube were 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. The operational code could employ an extra iteration here, but a simpler approach is simply to use the most-recent achieved rotator angle, extrapolated according the current velocity. Such difficulties occur only near the zenith (within say 10 arcminutes), where there are other problems, including the possibility of oscillation as rotator angle and telescope azimuth chase each other in order to maintain pointing. The operational code may need to employ brutal techniques to keep out of the region altogether.

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-origin 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 conceivably have a practical use: a detector placed near the edge of the $-0\overset{\circ}{.}75$ field might be able to track a region of sky right through the zenith if the full 2°/s azimuth slewing speed is available. 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 what is traditionally called the 'telescope tube' but which in the case of the Gemini design is the complex ensemble of the optical support structure (OSS), the instrument rotator, the acquisition and guiding unit and its components, together with the active optics system. Starting with the entire telescope unaffected by gravity, the next step is to imagine the gravity being 'turned on', potentially causing the OSS 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 OSS 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 was eventually 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 is likely to be extremely small for the Gemini design, because the active optics system will eliminate all the gross deformations in the OSS, just a few arcseconds at the most, and its empirical nature make it unnecessary to do other than an approximate inversion. Thus the provisional 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). Without active optics and with flexure alone affecting pointing, a downward droop of the top end 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 turn out to be 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 $[\,Az, El\,]$, and the result, `X2,Y2,Z2`, is the mount $[\,Az, El\,]$.

```
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 decision is ever made that the telescope tube be allowed 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 MCS/servo system to close the position loop.

For each of the two sets of encoders there will, of course, be a zero point, and we 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 the analysis of pointing tests it would also be tidy to arrange that the mechanical setting-up of that encoder system is also such that the index error *IA* is small and the encoder is consistent with the gross telescope. Note also that stability of the *IE* and *IA* values could be crucial if large and rapidly changing encoder 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 involved, 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 tracking tests instead, either specially-devised or by means of the analysis of logged WFS data. Though possibly hard to measure, such errors ought to be very stable.

We plan to introduce small random variations when setting the telescope to standard park positions. This will avoid localized wear as a result of movement during oilpad pressure changes *etc.*)

6 INSTRUMENT-MOUNT POSITION-ANGLE

Commands and other controls will be provided to allow data acquisition procedures and the observer to specify the orientation of the instrument-mount. The following functions will be supported:

- In the tracking reference frame, set the rotator y-axis to a given angle relative to the meridian which passes through the pointing origin. 'Meridian' means northwards for equatorial coordinates, up for $[\,Az, El\,]$.

- In the observed $[\,Az, El\,]$ frame (irrespective of the tracking frame), set the rotator y-axis to a given angle relative to the vertical which passes upwards through the pointing origin.

Either of the set functions should be available *via* direct pushbutton control of the rotator (so there will 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 (see the Starlink SLALIB routines PA and ALTAZ for example). However, the Gemini accuracy goals mean that refraction and at least some of the telescope pointing corrections must 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 altazimuth mounts, 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, because the telescope tracking calculations are being 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[3]:

1. Generate the '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 $[\,Az, El\,]$ 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 $[\,Az, El\,]$ 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.

[3]The approach described here was used in early versions of the Gemini TCS. A different, more versatile, algorithm was subsequently adopted, based on "osculating transformation matrices" (yet to be described). Details of this alternative algorithm are given later.

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[4] 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.0DOD.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.

(An impure but perhaps easier to understand version of the above algorithm, avoiding use of the 'up vector', would be to repeat the pointing calculations for an imaginary target a few arcseconds north of the true target and to use the resulting changes in $[\,Az, El\,]$ to determine the orientation of the north-south line.)

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 corrected position-angle is approximately:

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

[4] Mathematically, it turns out that the value of `ZU` is almost immaterial.

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 taking a CCD picture) is to specify a pointing-origin some distance off-centre and to make the target position a point on the sky about that distance from the pole and with α chosen to determine the orientation of the field on the detector.

7 PRACTICAL DETAILS

7.1 Pointing Adjustments

The horizontal collimation *CA* and elevation index error *IE* are of special importance as they cope with a wide range of physical effects which might otherwise pose an overwhelming calibration problem. Examples include unwanted tip/tilts in M1 and M2, and thermal distortions of the telescope tube.

For this reason, these two numbers, *CA* and *IE*, should be the ones adjusted when the telescope pointing is checked on a few stars at the start of observing.

Functions will be provided to set and calibrate *CA* and *IE*:

1. One function will simply report the current values and allow new ones to be entered.
2. Another will allow the telescope operator (or an automatic procedure involving a WFS) to signal that the telescope has been adjusted to point at the target whose coordinates were supplied. Appropriate changes can then be made to *CA* and *IE* so that the current and supplied target coordinates match.
3. A third function will manage a calibration run involving one or more stars. During the run, records will be kept of the stars used so far, and the mean and RMS for the corrections applied to *CA* and *IE* will be logged as an indication of pointing stability. As more stars are observed, it will become possible to adjust other parts of the telescope pointing model which are known to vary from day to day. The instrument rotator will have to be kept stationary (*i.e.* with respect to the OSS) during the procedure, unless the pointing-origin x, y relative to the rotator axis is accurately known.

7.2 Calibrating the Pointing-Origin Positions

The online measurement and adjustment of the position XIM,YIM of each pointing-origin 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 *IE* values to correct the pointing locally for example).

Provision for several pointing-origins will be made. Three will be enough for most purposes (two instrument apertures plus a nominal origin for acquisition and field verification) but more will be needed in cases such as M2 chopping with a multi-aperture instrument. Manual operation through the TCS console might allow the user to name a separate command for each required origin, 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
```

Selection of pointing-origin *via* pushbutton may also be useful. Control from the OCS may involve either selection of preset pointing-origins or simply assertion of new XIM,YIM values.

The pointing-origin calibration procedure will poll a set of keys or pushbuttons (for example a hand paddle: up/down/left/right plus speed control are required) and will directly increment and decrement XIM and YIM. The operator will first position a star on a known and accurately calibrated origin (probably a TV reference point), then will select the new origin (by pressing the 'aperture A' button, say) and then use 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.

Calibration with respect to the rotator axis will be performed by determining the XIM,YIM of any given origin twice with a 180° rotation in between. An adjustment of half the difference between the two sets of XIM,YIM is then made to all the pointing-origins, which are then referred to the rotator axis.

Pressing the buttons while the standard 'reference origin' (used for target acquisition) is selected will have the effect of changing the telescope demand $[\alpha, \delta]$ while making equal and opposite changes in XIM,YIM for all the other pointing-origins. This is to allow any tracking errors to be eliminated during the calibration run.

Facilities will be required for automatically centring on the source. The procedure will involve sampling the detector output while performing a cross-scan. Each arm of the scan will be self-convolved to determine the point of best symmetry (this will work even if the signal is negative-going), and the pointing-origin XIM,YIM will then be adjusted so that the telescope moves to align the beam to the source.

7.3 Economical Implementation

There have been proposals to carry out the full pointing calculation at or near servo rates; the Gemini IOCs are very fast compared with the telescope control computers of a few years ago, and the resulting 'brute force' software will be as compact and obvious as it is possible to achieve. However, there are counter-arguments: computing power is still not in infinite supply, 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 thrift now leaves open the option later to increase the loop speed from the planned 20 Hz if this turns out to be desirable.

A simple scheme was described earlier for computing the sidereal time, essentially by counting 20 Hz "ticks". In the case of the pointing transformations, we intend to use a scheme which combines rigour 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:

1. At **low** frequency; about once every 60^s 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?
2. At **medium** frequency; limited by how far the telescope can offset between iterations, and hence how far out the pointing predictions will be – about once every 5^s is satisfactory:
    - Generate 1st OTM: mean $[\,\alpha,\delta\,]$ to apparent $[\,\alpha,\delta\,]$
    - Generate 2nd OTM: apparent $[-h,\delta\,]$ to pre-collimation $[\,Az,El\,]$
    - amount of tube flexure if complicated function
3. 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 $[\,Az,El\,]$ (using 2nd OTM)
    - collimation
    - to mount $[\,Az,El\,]$
    - 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-origin x, y parameters). Figure 3 shows the relationship between the two OTMs and the 'pointing flow'.

There will be analogous but hopefully much simpler transformations to perform in order to generate corrected encoder readings. It seems likely that the corrections will change sufficiently slowly for them to be computed at a much lower frequency than reading the encoders and issuing the corrected reading to the servo software. The computations can take place in either the TCS or the MCS. 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. We propose using an equilateral triangle about $0^\circ\!\!.5$ a side, 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 modelled.

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:

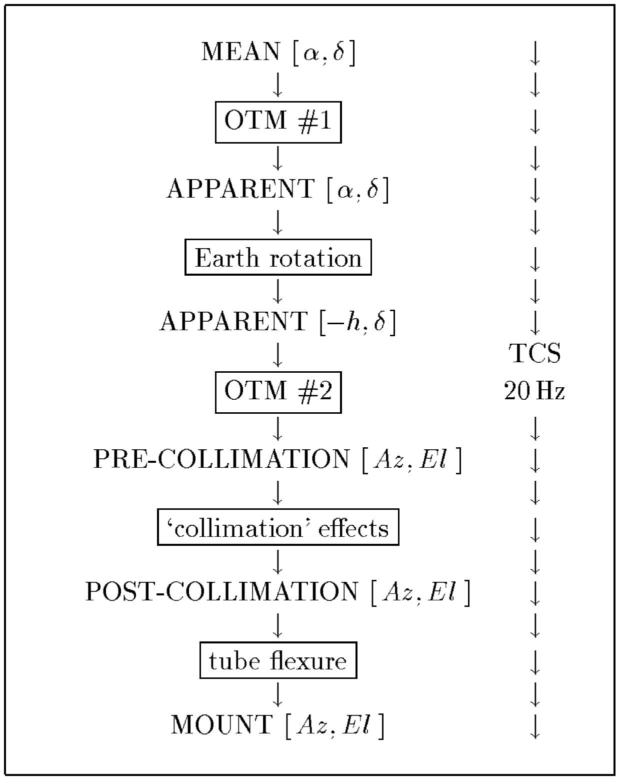

Figure 3: **The TCS Pointing Flow Using OTMs**

The diagram is adapted from the top (TCS) part of Figure 1, but with each of two groups of transformations summarized into a 3×3 matrix, called an 'osculating transformation matrix' or OTM. In cases where the telescope is being controlled in terms of apparent [α, δ] or topocentric [Az, El], there is no OTM #1, and if control is in mount [Az, El] neither OTM is required.

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

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

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

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 rather 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 simple 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 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″) gives good results, as will a wide range of other values. The pattern is a fairly accurate equilateral triangle centred on the target, but this is similarly uncritical. We will, however, 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)
```

The inverse of each OTM will also be required, for performing the 'upstream' pointing flow. The upstream flow, which starts with mount $[\,Az, El\,]$ and calculates from it the corresponding celestial position, is the subject of the next section.

7.4 Upstream Transformations

The principal concern of the present document is open-loop control of the telescope, and so we have concentrated on a 'pointing flow' which begins with a star's catalogue position and predicts what encoder readings are required in order to point at the star. It is, however, also possible to formulate a pointing flow which works in the opposite direction, and the TCS will need to do this for some purposes. We call the star-to-mount, or "Where should I point the telescope?" case the *downstream* transformation, and the mount-to-star, or "Where is the telescope pointed?" case the *upstream* transformation.

An obvious example requiring an upstream transformation is where the telescope is on its way to a new target and we would like to monitor its actual position *en route*. However, much more critical applications of upstream transformations arise in the management of autoguiding, for example when determining the $[\alpha, \delta]$ of a guide-star (in the usual case, where only an approximate position is available *a priori*.)

When the telescope is a long way from the target (the point on the sky for which accurate pointing calculations are being made) the upstream transformation will use simple, classical transformations, and ignore pointing corrections. Once the telescope is within a few degrees of the target, and approaches the region which is being sampled to provide the OTMs, more elaborate calculations (see below) can take over.

On some occasions an upstream pointing flow will be required only for a segment of the full, downstream pointing flow. However, we give here all the steps, starting with corrected encoder readings from the MCS and determining the corresponding mean $[\alpha, \delta]$ in the tracking coordinate system. (It should be noted that the variable-names used here are not necessarily the same as the ones used for the downstream code fragments.)

The first step is to transform mount $[Az, El]$ from spherical coordinates `AZ` and `EL` to Cartesian coordinates `XF`, `YF`, `ZF`:

```
*  Azimuth (N=0,E=90) and elevation to x,y,z
      COSEL = COS(EL)
      XF = -COS(AZ)*COSEL
      YF = SIN(AZ)*COSEL
      ZF = SIN(EL)
```

Then the effects of 'tube flexure' are removed. For consistency with the downstream alogorithm, this involves one Newton-Raphson iteration (written out). The tube flexure coefficient is `TF`, and the resulting $[x, y, z]$ coordinates are `XC`, `YC`, `ZC`:

```
*  Remove tube flexure
      Z = (ZF-TF)*(1D0+TF*ZF)
      ZC = (ZF-TF*(1D0+Z*Z))/(1D0-2D0*TF*Z-TF*TF)
      F = 1D0-TF*ZC
      XC = XF/F
      YC = YF/F
```

The 'collimation' terms `XI` and `ETA` are then established; these are the so-called 'standard coordinates' ξ and η. Variables `XIM` and `YIM` are the $[x, y]$ of the nominated pointing-origin; `CA` and `CE` are the horizontal and vertical collimation terms. Note that because the nonperpendicularity effect (coefficient `NPAE`) depends on the mechanical rather than the pre-collimation elevation, it is simpler in form here than in the more awkward downstream case. Note also that allowance should be made at this stage for any optical distortion, relative to the ideal gnomonic projection.[5] The variable FL is the telescope focal length; SPA and CPA are the sine and cosine of the actual rotator position-angle.

```
*  Pointing axis position wrt rotator axis
      XR = XIM/FL
      YR = YIM/FL

*  Pointing axis position wrt ideal axis
      XI = -XR*CPA-YR*SPA+CA+NPAE*ZF
      ETA = XR*SPA-YR*CPA+CE
```

The collimation effects can then be allowed for by means of the following rigorous transformation; the resulting $[x, y, z]$ coordinates are XM, YM, ZM:

```
*  Remove combined collimation
      F = SQRT(1D0+XI*XI+ETA*ETA)
      R = SQRT(XC*XC+YC*YC)
      XM = (XC-(XI*YC+ETA*XC*ZC)/R)/F
      YM = (YC+(XI*XC-ETA*YC*ZC)/R)/F
      ZM = (ZC+ETA*R)/F
```

The second OTM's inverse OTMHEI is now used to transform the telescope position into local equatorial coordinates XH, YH, ZH:

```
*  To -HA/Dec system
      XH = OTMHEI(1,1)*XM+OTMHEI(1,2)*YM+OTMHEI(1,3)*ZM
      YH = OTMHEI(2,1)*XM+OTMHEI(2,2)*YM+OTMHEI(2,3)*ZM
      ZH = OTMHEI(3,1)*XM+OTMHEI(3,2)*YM+OTMHEI(3,3)*ZM
```

The coordinates are then referred to the true equinox by allowing for Earth rotation, to give $[x, y, z]$ coordinates XA, YA, ZA; SST and CCT are the sine and cosine of the local apparent sidereal time:

```
*  To geocentric apparent RA/Dec system
      XA = CST*XH-SST*YH
      YA = SST*XH+CST*YH
      ZA = ZH
```

The complexities of the transformation into, say, B1975 FK4 coordinates are encapsulated in the first OTM's inverse OTMUAI, which is applied to yield $[x, y, z]$ coordinates XU, YU and ZU:

[5] For the pure Ritchey-Chrétien optical design used by Gemini, and the relatively small field, the departure from tangent plane geometry is small enough to ignore.

```
*  To current tracking mean RA/Dec system
      XU = OTMUAI(1,1)*XA+OTMUAI(1,2)*YA+OTMUAI(1,3)*ZA
      YU = OTMUAI(2,1)*XA+OTMUAI(2,2)*YA+OTMUAI(2,3)*ZA
      ZU = OTMUAI(3,1)*XA+OTMUAI(3,2)*YA+OTMUAI(3,3)*ZA
```

Finally we transform from $[x, y, z]$ to $[\alpha, \delta]$:

```
*  To RA/Dec
      RA = ATAN2(YU,XU)
      DEC = ATAN2(ZU,SQRT(XU*XU+YU*YU))
```

For pointing coefficients within the ranges expected for the Gemini telescopes, consistency between the downstream and upstream algorithms presented in this paper is usually better than 10^{-5} arcseconds, even for pointing-origins lying well off-axis.

7.5 Instrument-Mount Position-Angle using OTMs

The algorithm presented in Section 6 expresses the orientation between (i) the $[x, y]$ coordinate system of the focal-plane and (ii) the celestial coordinate system in which the telescope is being controlled. A more general problem is to determine the focal-plane orientation with respect to an arbitrary celestial coordinate system which is in general different from the one in which the target is being tracked. The most obvious case is where a given instrument orientation with respect to $[Az, El]$ is required; merely stopping the rotator is, after all, only an approximation to this condition (and only works for an altazimuth mount). A more subtle example is where the telescope is being controlled in (say) J2000 $[\alpha, \delta]$ and it is important (for some reason) to have a 2D detector aligned accurately to the B1950 grid.

One way to implement the general scheme is to use OTMs which apply to the nominated rotator-tracking frame, separate from the ones which apply to the target-tracking frame. Upstream transformations can then be used to determine the orientation in the rotator-tracking frame of the focal-plane's η-axis, and hence the rotator orientation that aligns the η-axis (at the pointing-origin) with the "meridian" (the north-south or up-down line) in the rotator frame. The following code demonstrates this technique. Prerequisites are:

- the pre-collimation mount coordinates, `XM,YM,ZM`;
- the two OTMs (as in the upstream-transformation code, these are called `OTMHEI` and `OTMUAI`, though they may not contain the same numbers);
- `S` and `C`, the sine and cosine of the angle of rotation required between the application of the two OTMS, normally the sidereal time;
- the pointing-origin coordinates, `XI,ETA`; and
- the pre-tube-flexure mount coordinates, `XC,YC,ZC`.

```
*  Undo 2nd transformation (giving -HA/Dec in the usual case)
      XHI = OTMHEI(1,1)*XM+OTMHEI(1,2)*YM+OTMHEI(1,3)*ZM
      YHI = OTMHEI(2,1)*XM+OTMHEI(2,2)*YM+OTMHEI(2,3)*ZM
      ZHI = OTMHEI(3,1)*XM+OTMHEI(3,2)*YM+OTMHEI(3,3)*ZM

*  Undo rotation (giving apparent RA/Dec in the usual case)
      XAI = C*XHI-S*YHI
      YAI = S*XHI+C*YHI
      ZAI = ZHI

*  Undo 1st transformation (giving rotator tracking coordinates)
      XO = OTMUAI(1,1)*XAI+OTMUAI(1,2)*YAI+OTMUAI(1,3)*ZAI
      YO = OTMUAI(2,1)*XAI+OTMUAI(2,2)*YAI+OTMUAI(2,3)*ZAI
      ZO = OTMUAI(3,1)*XAI+OTMUAI(3,2)*YAI+OTMUAI(3,3)*ZAI

*  This is the projection on the sky of the pointing-origin
*  at XI,ETA, in the rotator-tracking frame.

*  Add an arbitrary eta increment to the pointing axis position
      ETA = ETA+1D0

*  Remove combined collimation
      R = MAX(SQRT(XC*XC+YC*YC),1D-10)
      XMI = XC-(XI*YC+ETA*XC*ZC)/R
      YMI = YC+(XI*XC-ETA*YC*ZC)/R
      ZMI = ZC+ETA*R

*  Undo 2nd transformation (giving -HA/Dec in the usual case)
      XHI = OTMHEI(1,1)*XMI+OTMHEI(1,2)*YMI+OTMHEI(1,3)*ZMI
      YHI = OTMHEI(2,1)*XMI+OTMHEI(2,2)*YMI+OTMHEI(2,3)*ZMI
      ZHI = OTMHEI(3,1)*XMI+OTMHEI(3,2)*YMI+OTMHEI(3,3)*ZMI

*  Undo rotation (giving apparent RA/Dec in the usual case)
      XAI = C*XHI-S*YHI
      YAI = S*XHI+C*YHI
      ZAI = ZHI

*  Undo 1st transformation (giving rotator-tracking coordinates)
      X = OTMUAI(1,1)*XAI+
          OTMUAI(1,2)*YAI+
          OTMUAI(1,3)*ZAI
      Y = OTMUAI(2,1)*XAI+
          OTMUAI(2,2)*YAI+
          OTMUAI(2,3)*ZAI
      Z = OTMUAI(3,1)*XAI+
          OTMUAI(3,2)*YAI+
          OTMUAI(3,3)*ZAI
```

```
*  Position angle of +eta on the rotator tracking frame
      SQ = -Y*XO+X*YO
      CQ = Z*(XO*XO+YO*YO)-ZO*(X*XO+Y*YO)
      IF (SQ.EQ.0D0.AND.CQ.EQ.0D0) CQ=1D0
      QC = ATAN2(SQ,CQ)
```

Notes:

1. The algorithm presented earlier involved a “correction to position-angle due to collimation error”. The present algorithm does not.
2. In the presence of refraction, any instrument-mount orientation is a compromise, and stars somewhere in the field will inevitably produce “trailed” images in a long exposure. Compared with the earlier algorithm, the values returned by the code above are slightly different but amount to an equally good compromise.

7.6 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-specified.

The external data are as follows:

- TAI
- telescope longitude, latitude, and height
- polar motion x, y
- TAI−UTC
- UT1−UTC
- TT−TAI ($= 32^{\mathrm{s}}.184$)

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
- AN: azimuth axis tilt, north
- AW: azimuth axis tilt, west
- $NPAE$: Az/El nonperpendicularity
- CA: horizontal collimation
- CE: vertical collimation

- focal length
- rotator position-angle, actual
- pointing-origin 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-origin selection
- required orientation of rotator y-axis