Tracking errors due to polar misalignment of EQ mounts

We will use Cartesian coordinate systems for the transformation from a basis that is fixed relative to the sky, to telescope coordinates. The transformations between them will be defined by orthogonal matrices to simplify the math. For convenience, assuming orthogonal base vectors, we refer to the bases by orthogonal matrices that have the XYZ base vectors as columns. We use subscripts for these matrices to refer to the various bases and the transformations between them.

Going from one orthogonal basis to the next changes the coordinates as follows. If the base vectors are the columns of a matrix M then any vector v is represented as $v = Mm$ where m contains the coordinates in that basis. If we switch to another basis that are the columns of a matrix N defined as $N = MT$ where T is orthogonal, then $v = Mm = MTT'm = Nn$ where the coordinates of v in N are $n = T'm$. Our coordinate transformations will all be composed from basic Given's rotations.

An appropriate representation of the night sky is the unit sphere with an infinitesimally small Earth at its center. For small angles, Euclidian distances on the unit sphere correspond with radians, which is convenient. For lack of a better term, let us refer to the Cartesian coordinate system that is fixed relative to the sky and where the Z axis is pointing to the NCP[1], as the universe coordinate system. Where the X and Y axes are pointing can be chose arbitrarily so long as the XYZ system is right-handed.

The subsequent bases for going from universe to telescope coordinates are as follows:

- **Universe:** Assume that Earth is at a fixed point in time t, say, $t = 0$ when we start observing and perform our polar alignment. Our first basis M_1 is chosen such that the Z axis points to the NCP, XYZ is right-handed and the observer's location is on the XZ plane.
- **Observer on Earth:** Our second basis M_2 rotates M_1 around the Y axis over an angle $\lambda_c = \frac{\pi}{2} - \lambda$ where λ is the latitude of the observer in radians. The direction of this rotation is such that the positive Z axis rotates towards the positive X axis. In this basis, Z points at zenith in the observer's coordinate system and the XY plane is aligned with the horizon. The negative X axis points North in the YZ plane below the NCP, and the positive Y axis points East. Let this transformation be defined by $M_2 = M_1 T_2$.
- **Earth's rotation:** Our third basis M_3 rotates M_2 around the Z axis of M_1 (points at the NCP) Eastward (positive X axis moves towards positive Y axis) in function of time at an angle ωt where ω is Earth's angular rotation velocity relative to the universe. This is suitable for describing how we experience the skies from our observing location at any point in time. Let this transformation be defined by $M_3 = M_2 T_3$.
- **Polar (mis)alignment:** Our fourth basis M_4 is that of the mounted telescope with Z defined as the RA axis of the mount. For this, we first rotate M_3 around the Y axis back over an angle $\lambda_c + \xi$ then around the rotated X axis of M_4 over an angle η (the positive direction is from the positive Z axis to the positive Y axis), where ξ and η represent the polar misalignment errors.

[1] Sorry, we will assume the Northern hemisphere only and not consider the SCP and East/West orientations for the Southern hemisphere.

Note that these angles are Euler angles defined by chained rotations, not polar coordinates. Let this transformation be defined by $M_4 = M_3 T_4$.

- **RA axis rotation:** Our fifth basis M_5 is where we rotate M_4 around its Z axis over an angle ωt Westward. This represents the tracking of the telescope just using its RA motor. Let this transformation be defined by $M_5 = M_4 T_5$.

To find the effect of polar misalignment of an object in the sky we then simply transform its coordinates from M_1 to $M_5 = M_1 T_2 T_3 T_4 T_5$ and study its behavior in function of time. The rotation matrices are as follows:

$$T_2 = \begin{pmatrix} \cos\lambda_c & 0 & \sin\lambda_c \\ 0 & 1 & 0 \\ -\sin\lambda_c & 0 & \cos\lambda_c \end{pmatrix}$$

$$T_3 = \begin{pmatrix} \cos\lambda_c & 0 & -\sin\lambda_c \\ 0 & 1 & 0 \\ \sin\lambda_c & 0 & \cos\lambda_c \end{pmatrix} \begin{pmatrix} \cos\omega t & -\sin\omega t & 0 \\ \sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\lambda_c & 0 & \sin\lambda_c \\ 0 & 1 & 0 \\ -\sin\lambda_c & 0 & \cos\lambda_c \end{pmatrix}$$

$$T_4 = \begin{pmatrix} \cos\lambda_c & 0 & -\sin\lambda_c \\ 0 & 1 & 0 \\ \sin\lambda_c & 0 & \cos\lambda_c \end{pmatrix} \begin{pmatrix} \cos\xi & 0 & -\sin\xi \\ 0 & 1 & 0 \\ \sin\xi & 0 & \cos\xi \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & \sin\eta \\ 0 & -\sin\eta & \cos\eta \end{pmatrix}$$

$$T_5 = \begin{pmatrix} \cos\omega t & \sin\omega t & 0 \\ -\sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Positive ξ corresponds with the positive Z axis moving towards the negative X axis (moving North), and positive η corresponds with the Z axis moving towards the positive Y axis (moving East). The transformations from the M_1 (universe) basis to the M_5 (mount) basis is given by $T = T_2 T_3 T_4 T_5$, which simplifies to

$$T = \begin{pmatrix} \cos\omega t & -\sin\omega t & 0 \\ \sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\xi & 0 & -\sin\xi \\ 0 & 1 & 0 \\ \sin\xi & 0 & \cos\xi \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & \sin\eta \\ 0 & -\sin\eta & \cos\eta \end{pmatrix} \begin{pmatrix} \cos\omega t & \sin\omega t & 0 \\ -\sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The base transformation is $M_5 = M_1 T$ and the mount coordinates are T' times the universe coordinates. In the natural Cartesian M_1 basis we would have $M_1 = I$ and $M_5 = T$.

Note that when $\xi = \eta = 0$ we have perfect tracking thus $M_1 = M_5$ and $T = I$. Note also that the observer's latitude has completely dropped out of the result.

Thus, a target on the unit sphere in universe coordinates

$$\begin{pmatrix} x(\vartheta, \varphi) \\ y(\vartheta, \varphi) \\ z(\vartheta, \varphi) \end{pmatrix} = \begin{pmatrix} \sin\vartheta\cos\varphi \\ \sin\vartheta\sin\varphi \\ \cos\vartheta \end{pmatrix}$$

has mount coordinates

$$P(\vartheta,\varphi,t,\xi,\eta) = T(\omega,t,\xi,\eta)' \begin{pmatrix} x(\vartheta,\varphi) \\ y(\vartheta,\varphi) \\ z(\vartheta,\varphi) \end{pmatrix}$$

Test points for a camera

Let the target have universe coordinates (ϑ_0, φ_0):

$$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} = \begin{pmatrix} \sin\vartheta_0 \cos\varphi_0 \\ \sin\vartheta_0 \sin\varphi_0 \\ \cos\vartheta_0 \end{pmatrix}$$

Note that we can write this as an Euler rotation of the Z axis unit vector of the mount coordinate system

$$\begin{pmatrix} \sin\vartheta_0 \cos\varphi_0 \\ \sin\vartheta_0 \sin\varphi_0 \\ \cos\vartheta_0{}' \end{pmatrix} = \begin{pmatrix} \cos\varphi_0 & -\sin\varphi_0 & 0 \\ \sin\varphi_0 & \cos\varphi_0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\vartheta_0 & 0 & \sin\vartheta_0 \\ 0 & 1 & 0 \\ -\sin\vartheta_0 & 0 & \cos\vartheta_0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

For a camera sensor of size $w \times h$ and a telescope of focal length F, Let us define the projection of the corners of the camera sensor around the Z axis on the unit sphere as

$$\frac{1}{2F} \begin{pmatrix} \pm h \\ \pm w \\ 1 \end{pmatrix} \Bigg/ \left\| \begin{pmatrix} h \\ w \\ 1 \end{pmatrix} \right\|$$

We can then apply the same transformation to the corners to center the camera's FOV around the target. Applying $P(\vartheta,\varphi,t,\xi,\eta)$ to these test points yields the star trails from $t = 0$ to $t = T$. After this we convert the result back to universe coordinates by pre-multiplying the trails with $T(\omega,0,\xi,\eta)$ that also make the starting points of the trails coincide for each test point. We then flip the result back up to the tangent plane of the unit sphere at the Z axis by applying the transpose (inverse) of the above transformation, and select the Y coordinate as the X coordinate in 2D, and minus the X coordinate as the Y coordinate in 2D. Finally, we plot the 2D trails and calculate their statistics over the set of misalignment angles.

Drift and field rotation

Let (ϑ,φ) be the spherical coordinates of any point of interest, for instance the target or the corners of the camera's FOV. The uncontrolled drift of the object from $t = 0$ to $t = T$, using the start and end points only, is then defined as

$$D(\vartheta,\varphi,T,\xi,\eta) = P(\vartheta,\varphi,T,\xi,\eta) - P(\vartheta,\varphi,0,\xi,\eta)$$

The term uncontrolled indicates that no control is applied to keep the central object in the FOV in its place.

Let us consider the controlled case where the central object remains fixed by perfect tracking and/or autoguiding and where the only drift is field rotation in the corners of the FOV. We can create this situation by subtracting the drift of the target from the drift of all test points:

$$E(\vartheta, \varphi, t, \xi, \eta) = P(\vartheta, \varphi, t, \xi, \eta) - P(\vartheta_0, \varphi_0, t, \xi, \eta)$$

The field rotation is then described by how the difference between start and end point,

$$F(\vartheta, \varphi, T, \xi, \eta) = E(\vartheta, \varphi, T, \xi, \eta) - E(\vartheta, \varphi, 0, \xi, \eta)$$

These error trails are in 3D near the unit sphere. We can transform them to 2D trails in the same way as described in the previous section.

When people talk about polar misalignment, they usually mean the magnitude in no particular direction. The tracking errors depend significantly on that direction. Since it is not easy for a user to know in what direction the RA axis is misaligned, we can take the RMS over a number of polar misalignment angles $(\xi \quad \eta)$ as described above with a Euclidian norm equal to a given polar misalignment magnitude and return the RMS value of the tracking errors as our error measure.

Example

As an example, let us consider a telescope of 750 mm focal length with a 36x24 mm camera sensor. The observer is at $\lambda = 34$ degrees latitude and the target coordinates as seen by the observer are an altitude of $\alpha = 70$ degrees and an azimuth of $\zeta = 100$ degrees. The polar misalignment magnitude is $\delta = 2$ arc minutes and the exposure time is $T = 300$ seconds. How big are the star trails in the corners of the image (on average for all possible directions of polar misalignment)?

Our analysis requires the target location in the M_1 basis while our problem gives it in the M_2 basis, so first we have to transform it back. In the M_2 coordinate system we have spherical coordinates $\vartheta_2 = \frac{\pi}{180}(90 - \alpha)$ radians and $\varphi_2 = \frac{\pi}{180}(180 - \zeta)$ radians. Its Cartesian coordinates in the M_2 basis are

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} \sin\vartheta_2 \cos\varphi_2 \\ \sin\vartheta_2 \sin\varphi_2 \\ \cos\vartheta_2 \end{pmatrix}$$

The Cartesian coordinates in the M_1 basis are

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = T_2 \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} \sin\vartheta \cos\varphi \\ \sin\vartheta \sin\varphi \\ \cos\vartheta \end{pmatrix}$$

This also yields the spherical coordinates $(\vartheta \quad \varphi)$. Alternatively, we can simply assume that the equatorial Dec and HA coordinates are known and convert them to spherical coordinates, which is easier. We just wanted to illustrate a common setup for a user on the ground with his horizon and meridian.

We want to find how the polar misalignment affects the target and the corners of the camera's FOV centered around it. For this, note that the target is the result of two rotations acting on the NCP:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \sin\vartheta\cos\varphi \\ \sin\vartheta\sin\varphi \\ \cos\vartheta \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

To add the corners of the camera's FOV we can define a sensor projection on the unit circle around the NCP. Let w and h be the sensor width and height, respectively. If F is the focal length of the telescope, this corresponds with angular values $\frac{w}{F}$ and $\frac{h}{F}$ in a small-angle approximation. The matrix S defined as

$$S = \frac{1}{2F} \begin{pmatrix} -h & -h & h & h \\ -w & w & -w & w \\ 1 & 1 & 1 & 1 \end{pmatrix} \Bigg/ \left\| \begin{pmatrix} w \\ h \\ 1 \end{pmatrix} \right\|$$

has columns that define this projection centered around the Z axis with the long side parallel to the Y axis. The columns of the matrix

$$M = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{pmatrix} \left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} S \quad S \right)$$

are then the target and the corners of the camera's FOV centered around the target with the long side parallel to the XY plane. These are the points that we want to test for polar misalignment.

Note that we can write

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \sin\vartheta\cos\varphi \\ \sin\vartheta\sin\varphi \\ \cos\vartheta \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

On those corners, and the target itself, we can now run the error formula for a suitable range of values for $(\xi \quad \eta)$ and calculate the statistics. The results, uncontrolled, are shown below.

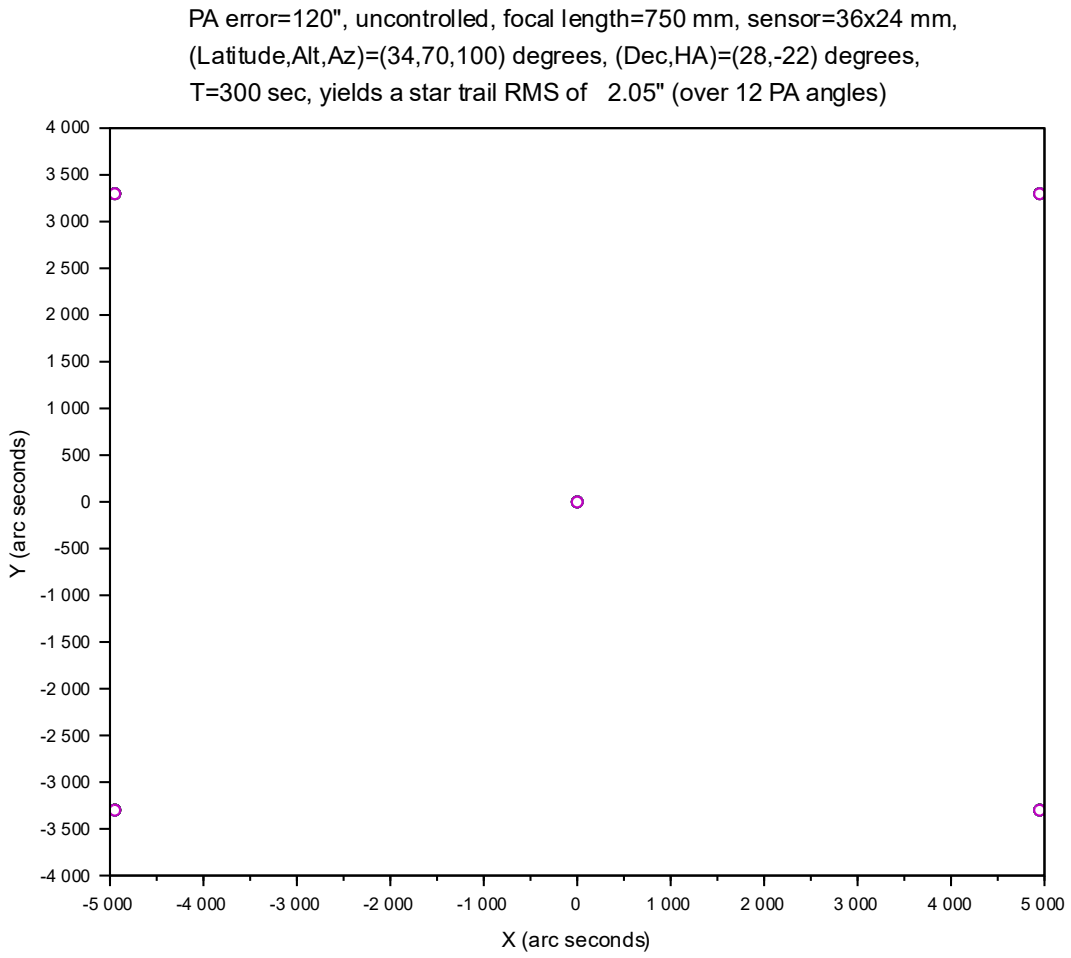
We have not discussed how to project the field rotation $F(\vartheta, \varphi, T, \xi, \eta)$ on the tangent plane to the unit sphere at (ϑ, φ). With the transformation above, this is the projection on the XY plane (also accounting for some sign and coordinate swaps of XY after rotating upwards):

$$\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\vartheta & 0 & \sin\vartheta \\ 0 & 1 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta \end{pmatrix} F(\vartheta, \varphi, T, \xi, \eta)$$

This corresponds with the errors displayed in the pictures that follow. For $D(\vartheta, \varphi, T, \xi, \eta)$ we do the same although the radial component is negligible relative to its size.

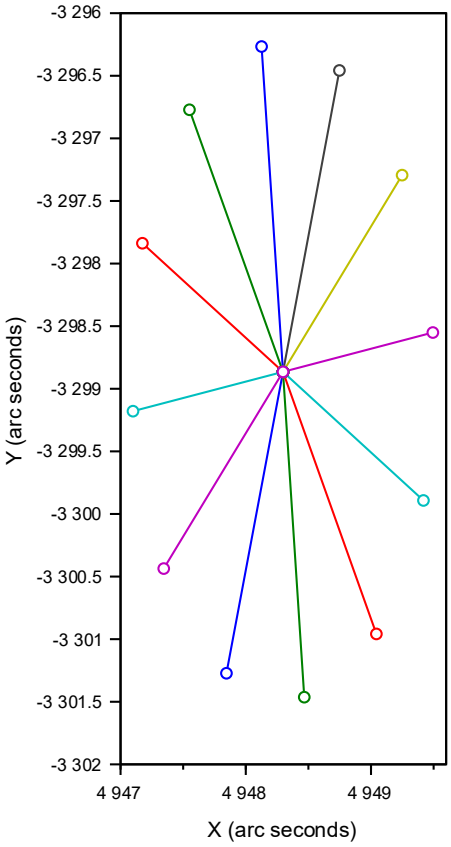
The first result is the uncontrolled case leading to an error RMS of 2.05”. The entire FOV is shown. The errors are visible and look similar at all 5 points. An RMS value of 2.05” over 5 minutes is 0.41” per minute, which is easy for an autoguider to correct. Still, alignment is recommended because you don’t want an autoguider to work harder than it should.

7

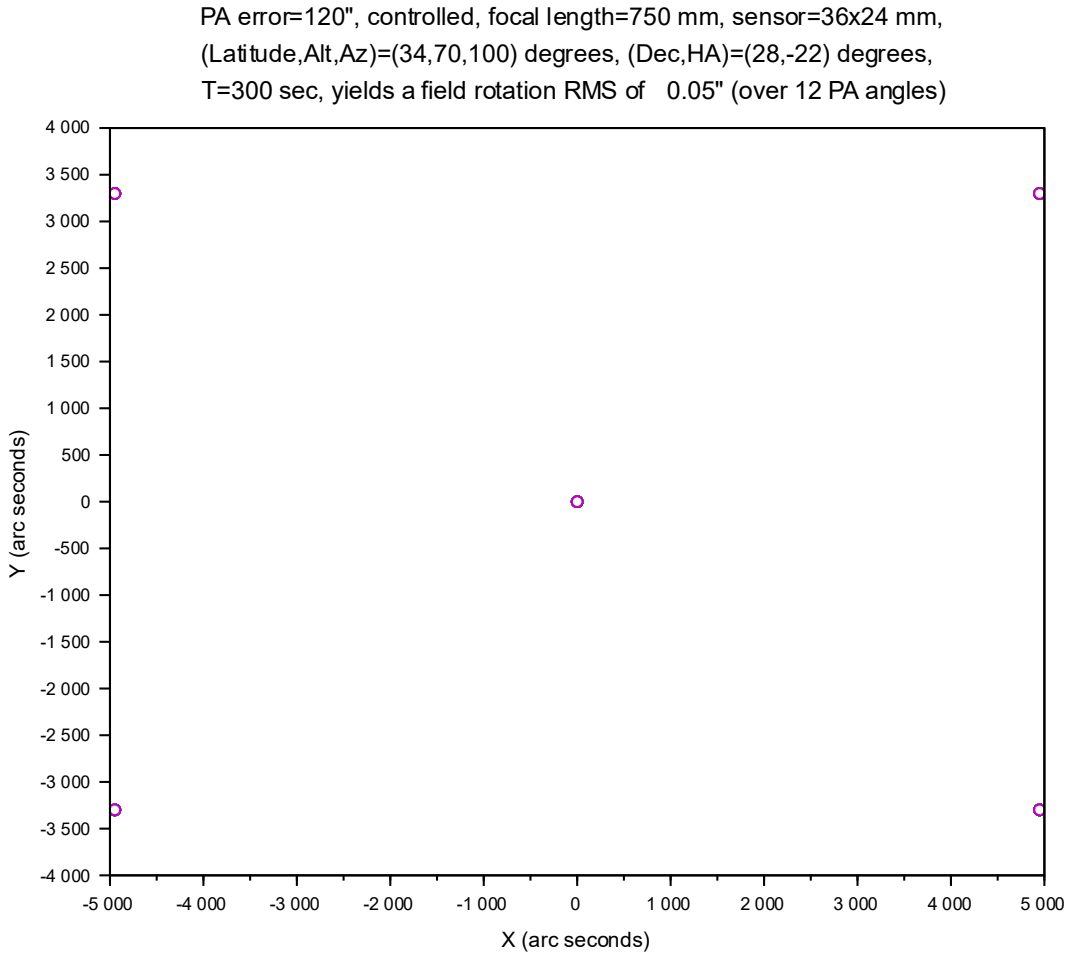


A zoomed-in version for the target (center of FOV) is shown below. Notice how much the tracking error varies across polar misalignment angles, and how large it is.

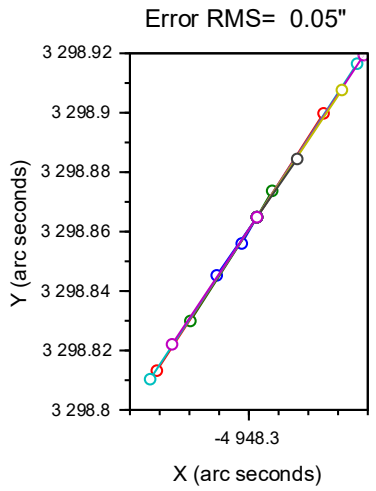
PA error=120", uncontrolled, focal length=750 mm, sensor=36x24 mm,
(Latitude,Alt,Az)=(34,70,100) degrees, (Dec,HA)=(28,-22) degrees,
T=300 sec, yields a star trail RMS of 2.04" (over 12 PA angles)

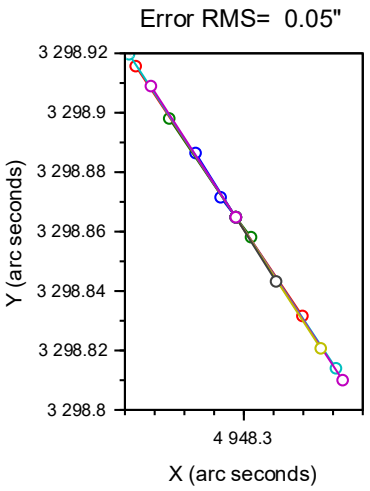


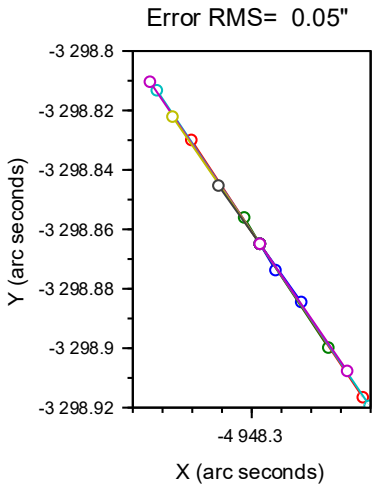
Below is a picture of the controlled situation over the entire FOV. Clearly the errors have been reduced drastically, to the point where one would have to zoom into the corners twice to spot any sign of field rotation. The tracking errors are now just due to field rotation, and are reduced from 114” to just 2.6”!

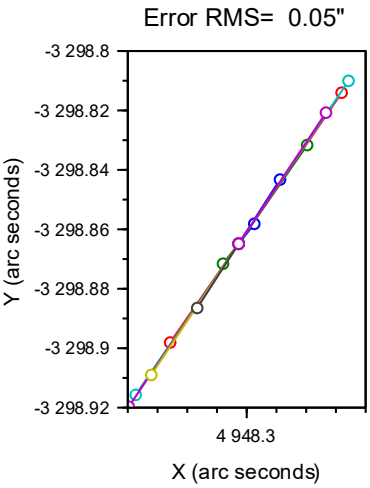


This is a zoomed-in view of the field rotation errors in the corners. The colors are overlapping, which makes it a bit hard to discern them but the direction shows how they rotate around the center.









Approximation for small polar misalignment angles

For small $(\xi \quad \eta)$ the approximation to first order is

$$
\begin{aligned}
M_5 &\approx \begin{pmatrix} \cos\omega t & -\sin\omega t & 0 \\ \sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & -\xi \\ 0 & 1 & 0 \\ \xi & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & \eta \\ 0 & -\eta & 1 \end{pmatrix}\begin{pmatrix} \cos\omega t & \sin\omega t & 0 \\ -\sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&\approx \begin{pmatrix} \cos\omega t & -\sin\omega t & 0 \\ \sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & -\xi \\ 0 & 1 & \eta \\ \xi & -\eta & 1 \end{pmatrix}\begin{pmatrix} \cos\omega t & \sin\omega t & 0 \\ -\sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} \cos\omega t & -\sin\omega t & 0 \\ \sin\omega t & \cos\omega t & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} \cos\omega t & \sin\omega t & -\xi \\ -\sin\omega t & \cos\omega t & \eta \\ \xi\cos\omega t + \eta\sin\omega t & \xi\sin\omega t - \eta\cos\omega t & 1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & -\xi\cos\omega t - \eta\sin\omega t \\ 0 & 1 & -\xi\sin\omega t + \eta\cos\omega t \\ \xi\cos\omega t + \eta\sin\omega t & \xi\sin\omega t - \eta\cos\omega t & 1 \end{pmatrix}
\end{aligned}
$$

Thus, the observed track $P(\vartheta, \varphi, \omega, \xi, \eta)$ is approximated by

$$
P(\vartheta, \varphi, t, \xi, \eta) = M_5 \begin{pmatrix} x \\ y \\ z \end{pmatrix} \approx \begin{pmatrix} x - z(\xi\cos\omega t + \eta\sin\omega t) \\ y + z(\eta\cos\omega t - \xi\sin\omega t) \\ z + (x\xi - y\eta)\cos\omega t + (x\eta + y\xi)\sin\omega t \end{pmatrix}
$$

For small polar misalignment angles this approximation is quite good, and it allows us to use analytical expressions that are not overly complex.

Relationship to other coordinate systems

We already mentioned the spherical coordinate system. It is defined by angles $(\vartheta \quad \varphi)$ where ϑ is the angle between a vector and the Z axis, and where φ is the angle between the X axis and the projection of a vector on the XY plane, positive in the direction where the positive X axis rotates towards the positive Y axis.

The equatorial coordinate system defines the declination (Dec) angle as $\frac{\pi}{2} - \vartheta$ in degrees, and the hour angle (HA) as $-\varphi$ if the X axis were pointing at the meridian, expressed in hours. The home position of equatorial mounts is defined by an HA value of 6 hours. This means that the RA axis must rotate 90 degrees or 6 hours towards the West to allow the Dec axis to rotate the telescope along the meridian. We will not need the definition of right ascension (RA) for our purposes and can avoid talking about equinoxes. It will be convenient to assume that the X axis is aligned with the meridian.

While we just concluded that the location of the observer is irrelevant, we had to define the coordinate system M_2 in which the observer's telescope is located, and talked about the Alt/Az coordinate system. The altitude (Alt) coordinate is the angle between the horizontal plane and the target. The azimuth (Az) is the angle between the projection of the NCP on the horizontal plane and the projection of the target

on the horizontal plane, positive in the direction from East to South. Both are expressed in degrees. Thus, Az=180 degrees corresponds with HA=0 hours (the meridian), and the positive direction of Az and HA is the same.

Scilab code

This code shows the effect of polar misalignment given the Alt/Az target at a given latitude. The parameters can be changed in the lines at the top. While it is not too difficult to change the code to start from HA/Dec coordinates, any planetarium program provides easy conversions to Alt/Az. Install Scilab, save this code as a .sce file, load it in the editor and hit F5.

```
d2r = %pi/180;                          // Degrees to radians
r2d = 1/d2r;                            // Radians to degrees
s2r = d2r/3600;                         // Arc seconds to radians
r2s = 1/s2r;                            // Radians to arc seconds

////////////////// USER PARAMETERS ///////////////////////
w = 36;                                 // Sensor width in mm
h = 24;                                 // Sensor height in mm
fl = 750;                               // Focal length
lambda = d2r*34;                        // Latitude in radians
delta = s2r*120;                        // Polar misalignment magnitude
alpha = d2r*70;                         // Target altitude in observer frame
zeta = d2r*100;                         // Target azimuth in observer frame
omega = 2*%pi/(24*3600);                // Earth's angular velocity
tau = 300;                              // Exposure time in seconds
npa = 12;                               // Number of polar misalignmeent angles
//////////////////////////////////////////////////////////

function [theta, phi]=c2s(xyz)
    // Cartesian to spherical coordinates
    // xyz can be a matrix where tje columns are the 3D vectors
    [m,n] = size(xyz);
    phi = zeros(n,1);
    for i = 1:n do
        x = xyz(1,i);
        y = xyz(2,i);
        z = xyz(3,i);
        theta = acos(z);
        if sin(theta) <> 0 then
            phi(i) = acos(x/sin(theta));
        else
            phi(i) = 0;
        end
    end
endfunction

function [xyz]=s2c(theta, phi)
    // Spherical to Cartesian coordinates
    // theta and phi can be vectors
    n = length(theta);
    for i = 1:n do
        xyz(:,i) = [sin(theta(i))*cos(phi(i)); sin(theta(i))*sin(phi(i)); cos(theta(i))];
    end
endfunction

function [T_ksi, T_eta]=paMisalignment(ksi, eta)
    // Polar misalignment from Euler angles
    // Rotate around Y (positive is from X+ to Z+)
    c = cos(ksi);
    s = sin(ksi);
    T_ksi = [c,0,-s; 0,1,0; s,0,c];
    // Rotate around X (positive is from Z+ to Y+)
```

```
    c = cos(eta);
    s = sin(eta);
    T_eta = [1,0,0; 0,c,s; 0,-s,c];
endfunction

function [T_theta, T_phi]=ncpToTarget(theta, phi)
    // Rotation matrices from [0;0;1] to target in spherical coordinates
    c = cos(theta);
    s = sin(theta);
    T_theta = [c,0,s; 0,1,0; -s,0,c];
    c = cos(phi);
    s = sin(phi);
    T_phi = [c,-s,0; s,c,0; 0,0,1];
endfunction

function [trail, trail2d]=trackingError(omega, tau, pa, theta0, phi0, w, h, fl, control)
    // trail: 4D matrix of trails on the unit sphere
    // trail2D: 4D matrix of trail projection on tangent plane to unit sphere
    // omega: Earth's angular velocity
    // tau: Time period (exposure time)
    // pa: Polar misalignment Euler angles
    // theta0, phi0: Spherical coordinates of target
    // w,h:  Camera sensor width and height
    // fl: Focal length of telescope
    // control: %t for exact target tracking otherwise %f
    t = [0; tau];                           // Time line (begin and end point are enough)
    nt = length(t);
    npa = size(pa, 1);
    trail = zeros(3, nt, 5, npa);
    trail2d = zeros(2, nt, 5, npa);

    // Corners in universe coordinates centered around the target
    dh = h/(2*fl);
    dw = w/(2*fl);
    corners = [-dh,-dh,dh,dh;-dw,dw,-dw,dw;1,1,1,1]/norm([dh;dw;1]);
    [T_theta, T_phi] = ncpToTarget(theta0, phi0);
    // Test points (target and FOV corners) in universe coordinates
    points = [s2c(theta0, phi0), T_phi*T_theta*corners];

    for ipa = 1:npa do
        // Transformation matrix from universe to mount at t=0
        [T_ksi, T_eta] = paMisalignment(pa(ipa,1), pa(ipa,2));
        T0 = T_ksi*T_eta;
        for i = 1:5 do
            for it = 1:nt do
                c = cos(omega*t(it));
                s = sin(omega*t(it));
                T_t = [c,s,0; -s,c,0; 0,0,1];
                T = T_t'*T_ksi*T_eta*T_t;
                if control then
                    trail(:,it,i,ipa) = T0*T'*points(:,i) - T0*T'*points(:,1);
                else
                    trail(:,it,i,ipa) = T0*T'*points(:,i);
                end
                // Rotate back to NCP then project on XY plane and swap 2D coordinates
                trail2d(:,it,i,ipa) = [0,1,0;-1,0,0]*T_theta'*T_phi'*trail(:,it,i,ipa);
             end
        end
    end
    if control then
        trail(:, :, 1, :) = 0
    end

endfunction

// Transformation matrix from universe to observer basis
// Observer is at latitude lambda on the meridian (XZ plane)
// Target is at (Alt, Az) from the observer's viewpoint
lambda_c = %pi/2 - lambda;
c = cos(lambda_c);
s = sin(lambda_c);
```

```
T_obs = [c,0,s; 0,1,0; -s,0,c];
// Calculate the spherical observer coordinates
theta2 = %pi/2 - alpha;             // Spherical coordinate theta is 90 - alt
phi2 = %pi - zeta;                  // Az increases from 0 at -X to 90 at Y
xyz2 = s2c(theta2, phi2);
// Transform to universe coordinates
xyz = T_obs*xyz2;
// Calculate the spherical universe coordinates
[theta0, phi0] = c2s(xyz);

pa = [0:npa-1]'*2*%pi/npa;          // Polar misalignment angles
pa = [sin(pa), cos(pa)]*delta;      // Matrix of (ksi, eta) Euler angles

// Uncontrolled tracking
[trail, trail2d] = trackingError(omega, tau, pa, theta0, phi0, w, h, fl, %f);

// Plot of target and FOV corner trails (uncontrolled).
fig1 = scf(1); clf();
var = 0;
nData = 0;
for i = 1:5 do
    x2d = zeros(2,npa);
    y2d = zeros(2,npa);
    for ipa = 1:npa do
        tmp = r2s*trail2d(:,:,i,ipa);
        x2d(:,ipa) = tmp(1,:)';
        y2d(:,ipa) = tmp(2,:)';
        // Statistics
        tmp = tmp(:,2) - tmp(:,1);
        var = var + tmp'*tmp;
        nData = nData + 1;
    end
    plot(x2d, y2d, 'o-');
end
xlabel("X (arc seconds)");
ylabel("Y (arc seconds)");
txt = sprintf("PA error=%d"", uncontrolled, focal length=%d mm, sensor=%dx%d mm,\n" + ...
    "(Latitude,Alt,Az)=(%d,%d,%d) degrees, (Dec,HA)=(%d,%d) degrees,\n" + ...
    "T=%d sec, yields a star trail RMS of %6.2f"" (over %d PA angles)", ...
    r2s*delta, fl, w, h, r2d*lambda, r2d*alpha, r2d*zeta, ...
    r2d*(%pi/2-theta0), -r2d*phi0, tau, sqrt(var/nData), npa);
title(txt);
fig1.children(1).isoview = "on";

// Plot of target trails (uncontrolled)
fig2 = scf(2); clf();
var = 0;
nData = 0;
x2d = zeros(2,npa);
y2d = zeros(2,npa);
for ipa = 1:npa do
    tmp = r2s*trail2d(:,:,i,ipa);
    x2d(:,ipa) = tmp(1,:)';
    y2d(:,ipa) = tmp(2,:)';
    // Statistics
    tmp = tmp(:,2) - tmp(:,1);
    var = var + tmp'*tmp;
    nData = nData + 1;
end
plot(x2d, y2d, 'o-');
xlabel("X (arc seconds)");
ylabel("Y (arc seconds)");
txt = sprintf("PA error=%d"", uncontrolled, focal length=%d mm, sensor=%dx%d mm,\n" + ...
    "(Latitude,Alt,Az)=(%d,%d,%d) degrees, (Dec,HA)=(%d,%d) degrees,\n" + ...
    "T=%d sec, yields a star trail RMS of %6.2f"" (over %d PA angles)", ...
    r2s*delta, fl, w, h, r2d*lambda, r2d*alpha, r2d*zeta, ...
    r2d*(%pi/2-theta0), -r2d*phi0, tau, sqrt(var/nData), npa);
title(txt);
fig2.children(1).isoview = "on";

// Controlled tracking
```

```
[trail, trail2d] = trackingError(omega, tau, pa, theta0, phi0, w, h, fl, %t);

// Plot of target and FOV corner trails (controlled).
fig3 = scf(3); clf();
var = 0;
nData = 0;
for i = 1:5 do
    x2d = zeros(2,npa);
    y2d = zeros(2,npa);
    for ipa = 1:npa do
        tmp = r2s*trail2d(:,:,i,ipa);
        x2d(:,ipa) = tmp(1,:)';
        y2d(:,ipa) = tmp(2,:)';
        // Statistics
        if i > 1 then
            tmp = tmp(:,2) - tmp(:,1);
            var = var + tmp'*tmp;
            nData = nData + 1;
        end
    end
    plot(x2d, y2d, 'o-');
end
xlabel("X (arc seconds)");
ylabel("Y (arc seconds)");
txt = sprintf("PA error=%d"", controlled, focal length=%d mm, sensor=%dx%d mm,\n" + ...
    "(Latitude,Alt,Az)=(%d,%d,%d) degrees, (Dec,HA)=(%d,%d) degrees,\n" + ...
    "T=%d sec, yields a field rotation RMS of %6.2f"" (over %d PA angles)", ...
    r2s*delta, fl, w, h, r2d*lambda, r2d*alpha, r2d*zeta, ...
    r2d*(%pi/2-theta0), -r2d*phi0, tau, sqrt(var/nData), npa);
title(txt);
fig3.children(1).isoview = "on";

// Plot of field rotation in FOV corners (controlled)
fig4 = scf(4); clf();
for i = 2:5 do
    subplot(2,2,i-1);
    x2d = zeros(2,npa);
    y2d = zeros(2,npa);
    for ipa = 1:npa do
        tmp = r2s*trail2d(:,:,i,ipa);
        x2d(:,ipa) = tmp(1,:)';
        y2d(:,ipa) = tmp(2,:)';
    end
    plot(x2d, y2d, 'o-');
    xlabel("X (arc seconds)");
    ylabel("Y (arc seconds)");
    txt = sprintf("Error RMS=%6.2f"", sqrt(var/nData));
    title(txt);
    fig4.children(1).isoview = "on";
end

// This is for debugging purposes.  Check if the data is
// approximately in a flat horizontal plane through [0;0;1].
// If not, the rotations are wrong.
[T_theta, T_phi] = ncpToTarget(theta0, phi0);
fig5 = scf(5); clf();
for i = 1:5 do
    x3d = zeros(2,npa);
    y3d = zeros(2,npa);
    z3d = zeros(2,npa);
    for ipa = 1:npa do
        tmp = trail(:,:,i,ipa);
        tmp = T_theta'*T_phi'*tmp;
        x3d(:,ipa) = tmp(1,:)';
        y3d(:,ipa) = tmp(2,:)';
        z3d(:,ipa) = tmp(3,:)';
    end
    e = param3d1(x3d, y3d, z3d, 'o-');
    e.foreground = color("red");
    e.mark_mode = "on";
```

```
end
xlabel("X (arc seconds)");
ylabel("Y (arc seconds)");
zlabel("Z (arc seconds)");
title(sprintf("Debug window to check the rotations", r2d*60*delta));
fig5.children(1).isoview = "on";
```