

Simulation of a Gyroscope

Introduction

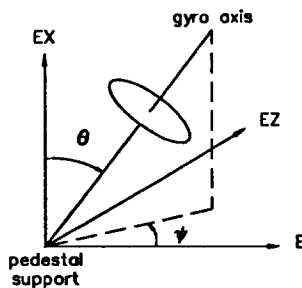
Gyroscopes fascinate us. We all remember spinning toy gyros as children. In their more sophisticated gimballed forms, gyros are also important technical devices, used, for example, for measuring motion in aerospace vehicles. ACSL simulation of gyro models can help you gain insight into the concepts and/or decide design criteria for actual applications.

Role of ACSL simulation

Simulation is an ideal way to describe a system, experiment with parameters, and check theories. ACSL makes the programming easy and encourages quick, interactive experiments.

Scope of model

The model described here is of a small free gyro, with a rotor inside a frame that rests on a pedestal. The gyro has its own axis frame, which is related to an inertial reference frame by Euler angles. The motion is started by pulling a string. The model includes friction of the rotor and of the spindle on the pedestal.



Equations of motion

The equations of motion of the gyro are derived from Newton's Laws. When a body rotates, the rate of change of angular momentum is equal to the applied torque. If angular momentum is H_G , the relationship can be written:

$$\frac{dH_G}{dt} = \text{torque}$$

Axis frames

To express this as components in a spinning frame, we use the transformation:

$$\frac{d}{dt} = (\dot{\quad})_G + \omega_G \times (\quad)_G$$

This says that the rate of change calculated in an inertial axis system is equal to the rate of change of the components in the spinning axis system together with the cross product of the axis spin vector ω_G and the components themselves. This formula applies to the components of any vector changing within the axis system itself, but we will apply it to the angular momentum vector, H_G .

The axis systems are shown in Figure 1. The G (gyro) axis system is non-rotating in inertial space about the gyro axis. There are no torques about this axis, although the gyro wheel itself is rotating and so there is an angular momentum about the axis. The turning rate of this axis system (ω_G) can be expressed as components in the G frame:

$$(\omega_{GX}, \omega_{GY}, \omega_{GZ})$$

Moment of inertia If I_X is the axial moment of inertia of the gyro, I_W is the axial moment of inertia of the wheel, and I_Y is the lateral moment of inertia about a line perpendicular to the gyro spin axis, then the angular momentum vector is:

$$(I_X \omega_{GX} + I_W \omega_X, I_Y \omega_{GY}, I_Y \omega_{GZ})$$

Angular momentum Because the angular rate ω_{GX} about the GX axis is zero, the momentum about GX can be replaced by H_X where:

$$H_X = I_W \omega_X$$

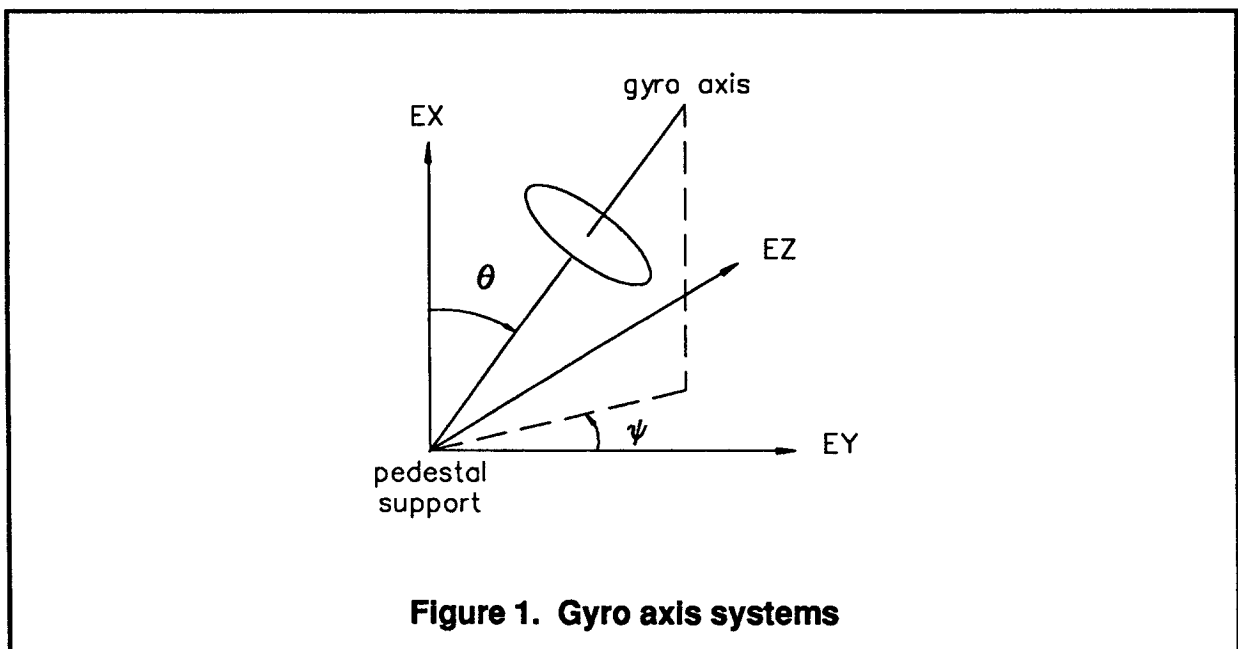
The total angular momentum vector then becomes:

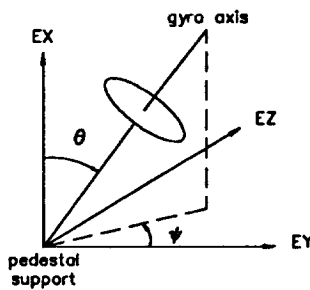
$$(H_X, I_Y \omega_{GX}, I_Y \omega_{GZ})$$

Using this in the equation above gives the GY and GZ components:

$$\frac{d}{dt}(I_Y \omega_{GY}) = I_Y \dot{\omega}_{GY} - H_X \omega_{GZ}$$

$$\frac{d}{dt}(I_Y \omega_{GZ}) = I_Y \dot{\omega}_{GZ} + H_X \omega_{GY}$$





Equating the inertial rates of change to the applied torques from the first equation above leads to the following:

$$I_Y \dot{\omega}_{GY} = T_{GY} + H_X \omega_{GZ}$$

$$I_Y \dot{\omega}_{GZ} = T_{GZ} - H_X \omega_{GY}$$

These equations are divided by the inertias so that only the accelerations ($\dot{\omega}_{GY}$ and $\dot{\omega}_{GZ}$) appear on the left side. The accelerations are then integrated directly to obtain ω_{GY} and ω_{GZ} .

Orienting the gyro in the inertial frame

With the angular rates known in an axis fixed in the gyro frame, we now orient the gyro relative to the base. The conventional gyro frame we have chosen is a yaw followed by a lean to line up with the gyro axis. To express this symbolically we first choose a base axis system in inertial (earth) coordinates (see Figure 1):

$$(EX, EY, EZ)$$

EX is vertical, positive upwards. EY is in the horizontal plane, visualized as out to the right. EZ completes the right hand set, forward in the horizontal plane, perpendicular to EY (EZ is EX crossed with EY).

Axis rotations

To move EX to the gyro axis, first rotate about a yaw angle ψ about EX and then lean over about the new EY through the lean angle θ until EX lines up with the gyro spin axis. The equations defining the angular rates of the orientation angles ψ and θ are as follows:

$$\dot{\psi} = -\omega_Y / \sin \theta$$

$$\dot{\theta} = \omega_Z$$

Because of the division by $\sin \theta$, these equations are invalid for a perfectly vertical gyro. If it is necessary to study gyro systems where the lean angle can become zero, then the orientation sequence must be reversed, *i.e.*, θ first about EZ, then ψ about the new EY.

Euler angles vs. quaternions

We chose the conventional gyro angles for ease of visualization and because the precession rate is calculated directly as $\dot{\psi}$, the yaw angle rate. In a system using Euler angles, a singularity shows up when the middle angle equals ninety degrees; *i.e.*, when the axis of the gyro is on or below the horizontal plane and the gyro falls off the pedestal. The Euler angle sequence for the angular rotations that describes the relative orientation of two axis systems always has a potential singularity. The more elegant solution, which gives full coverage of the sphere for all possible orientations, is to use quaternions.

Torque and friction

In this particular gyro, torque is produced by gravity acting on the center of mass and also by friction at the pedestal. The friction is assumed to be proportional to the turning rate at the pedestal. The gravity vector is:

$$(-g, 0, 0)$$

Gravity torque components are obtained by rotating the gravity vector into the G frame and calculating torques about GY and GZ. Since the yaw angle cannot affect the gravity components (the yaw axis is parallel to the gravity vector), the only

component is due to the center of gravity offset from the pedestal (dx_{cg}). Adding the gravity torque components and the torque components for damping due to friction leads to these torque equations:

$$\begin{aligned} T_{GY} &= -K_{GF} \omega_{GY} \\ T_{GZ} &= -K_{GF} \omega_{GZ} - mg dx_{cg} \sin \theta \end{aligned}$$

In gyros such as those used to measure missile turning rates, the designer tries to place the support as near to the center of gravity as possible since gravity-induced torques produce errors. Electronic torques are applied to keep the gyro centered in its casing by a feedback control system, and from these applied torques is obtained a measure of case spin rate.

Wind-up The gyro spin is started when the cord around the shaft is pulled. The equation for rotor speed is:

$$\dot{\omega}_X = f_s r_s / I_W \quad ; t < t_{windup}$$

where f_s is the force in the string (assumed about 30 N or 8 lb force) and r_s is the radius of the spindle (2 mm). After wind-up, the wheel slows down with a rate proportional to spin rate due to friction:

$$\dot{\omega}_X = -K_{WD} \omega_X \quad ; t > t_{windup}$$

Wind-up time The wind-up time is defined as the time during which the string is wrapped around the spindle. We can determine the amount of string unwrapped by:

$$\dot{l}_{pull} = \omega_X r_s$$

Startup continues while l_{pull} is less than the active length of the string wrapped around the spindle (about 0.5 m).

Simulation Program

The simulation program to model the gyro is shown in the listing of Figure 2. All units are in SI (meter, kilogram, second). ACSL keywords and operators (such as INTEG and CONSTANT) are shown in uppercase. The model variables are in lower case.

Wind-up A test for wind-up during the rotor acceleration is given by the logical expression:

$$\text{windup} = l_{pull} .LT. l_{string}$$

The expression thus is TRUE when the amount of string pulled off the spindle is less than the length of the string wrapped around the spindle.

Spin rate The rotor spin rate ω_X and the amount of string pulled are then obtained by:

$$\begin{aligned} wx &= \text{INTEG}(\text{RSW}(\text{windup}, f_s * r_s / i_w, -kwd * wx), wxz) \\ l_{pull} &= \text{INTEG}(\text{RSW}(\text{windup}, wx * r_s, 0.0), 0.0) \end{aligned}$$

```

PROGRAM simple gyro model

!-----define system constants
CINTERVAL cint = 0.010 ! data logging rate
ALGORITHM ialg = 4 ! integration algorithm (RK-2)
MAXTERVAL maxt = 0.0005 ! integration step size
NSTEPS nstp = 1
!-----model constants (SI units)
CONSTANT g = 9.81 ! acceleration of gravity (m/s2)
CONSTANT fs = 30 ! force in the string (N)
CONSTANT rs = 0.002 ! radius of the wheel spindle (m)
CONSTANT iw = 4.767e-5 ! inertia of gyro wheel (Kg m2)
CONSTANT iy = 4.300e-5 ! lateral inertia of gyro assembly
CONSTANT kwd = 0.02 ! wheel frictional coefficient
CONSTANT kgf = 0.004 ! pedestal frictional coefficient
CONSTANT mass = 0.085 ! mass of entire gyro assembly (Kg)
CONSTANT dxcg = 0.032 ! pedestal from center of gravity
CONSTANT lstring = 0.45 ! length of the pull string (m)
!-----initial values for the integrators
CONSTANT wxz = 0 ! initial wheel angular rate (r/s)
CONSTANT thz = 0.5 ! initial lean angle (r)
CONSTANT siz = 0 ! initial yaw angle (r)
CONSTANT wyz = 0 ! initial inertial rate about y
CONSTANT wzz = 0 ! initial inertial lean rate

LOGICAL windup
windup = lpull .LT. lstring
!-----wheel rate increases during windup, and
! then decays exponentially afterwards
wx = INTEG(RSW(windup, fs*rs/iw, -kwd*wx), wxz)
!-----the length of string pulled off the spindle
lpull = INTEG(RSW(windup, wx*rs, 0.0), 0.0)
!-----wheel angular momentum (Kg*m2/s)
hx = iw*wx
!-----torques about gyro center of gravity (Nm)
ty = -kgf*wy
tz = SIN(th)*mass*g*dxcg - kgf*wz
!-----angular accelerations, caged during windup
wyd = RSW(windup, 0.0, (ty - hx*wz)/iy)
wzd = RSW(windup, 0.0, (tz + hx*wy)/iy)
!-----angular velocities inside the gyro frame
! from the angular accelerations
wy = INTEG(wyd, wyz)
wz = INTEG(wzd, wzz)
!-----yaw angular velocity. note protection
! against the possible division by zero
sid = -wy/(SIN(th) + 1.0e-33)
!-----integrate for the two orientation angles
si = INTEG(sid, siz)
th = INTEG(wz, thz)
!-----projection of unit vector along axis on ground
y = SIN(th)*COS(si)
z = SIN(th)*SIN(si)
!-----termination conditions
CONSTANT tstop = 9.99 , thmx = 1.5
TERMT(t .GE. tstop, 'Stopped on time limit')
TERMT(th .GE. thmx, 'Stopped on excessive offset angle')
END ! of the PROGRAM

```

Figure 2. Gyro model ACSL source code

RSW RSW (real switch) is an ACSL function. If the first argument is TRUE, it returns the second argument; if the first argument is FALSE, it returns the third argument. The equation has the effect of accelerating the rotor spin only during wind-up.

Torque Gravity and friction torques about the gyro center of gravity are obtained from:

$$\begin{aligned}t_y &= -kgf*wy \\t_z &= \text{SIN}(th)*mass*g*dxcg - kgf*wz\end{aligned}$$

Angular accelerations The acceleration of the gyro is zero during wind-up of the rotor, representing the caging action of the person's hands. The angular accelerations in the gyro (G) frame can be written from the windup equations and gravity and friction torques equations using the RSW operator to test for wind-up.

$$\begin{aligned}wyd &= \text{RSW}(\text{windup}, 0.0, (t_y - hx*wz)/iy) \\wzd &= \text{RSW}(\text{windup}, 0.0, (t_z + hx*wy)/iy)\end{aligned}$$

Angular rates The angular rates for ψ and θ are calculated by the following equations. The division by $\text{SIN}(th)$ in the yaw rate equation is protected by the addition of a small term ($1.0e-33$) that prevents an overflow should θ ever become zero.

$$\begin{aligned}sid &= -wy/(\text{SIN}(th)+1.0e-33) \\thd &= wz\end{aligned}$$

Integrate for rates The rates are then integrated using the ACSL INTEG operator, which has arguments of derivative and initial condition:

$$\begin{aligned}si &= \text{INTEG}(sid, siz) \\th &= \text{INTEG}(wz, thz)\end{aligned}$$

Nutation In order to visualize the nutation, it is useful to plot the coordinates of the gyro tip as projected on the horizontal y-z plane. This is implemented using the projection of the unit vector through the gyro axes. These coordinates are:

$$\begin{aligned}y &= \sin \theta \cos \psi \\z &= \sin \theta \sin \psi\end{aligned}$$

Stopping the run The simulation stopping conditions are set using the TERMT operator. This model is set to stop at a given time (t_{stop}) or if the lean angle exceeds a maximum (th_{mx}). Each test may have a message associated with it. The message is written out when the run terminates on that limit.

```
CONSTANT tstop=9.99,thmx=1.5
TERMT(t.ge.tstop, 'Stopped on time limit')
TERMT(th.ge.thmx, 'Stopped on excessive offset angle')
```

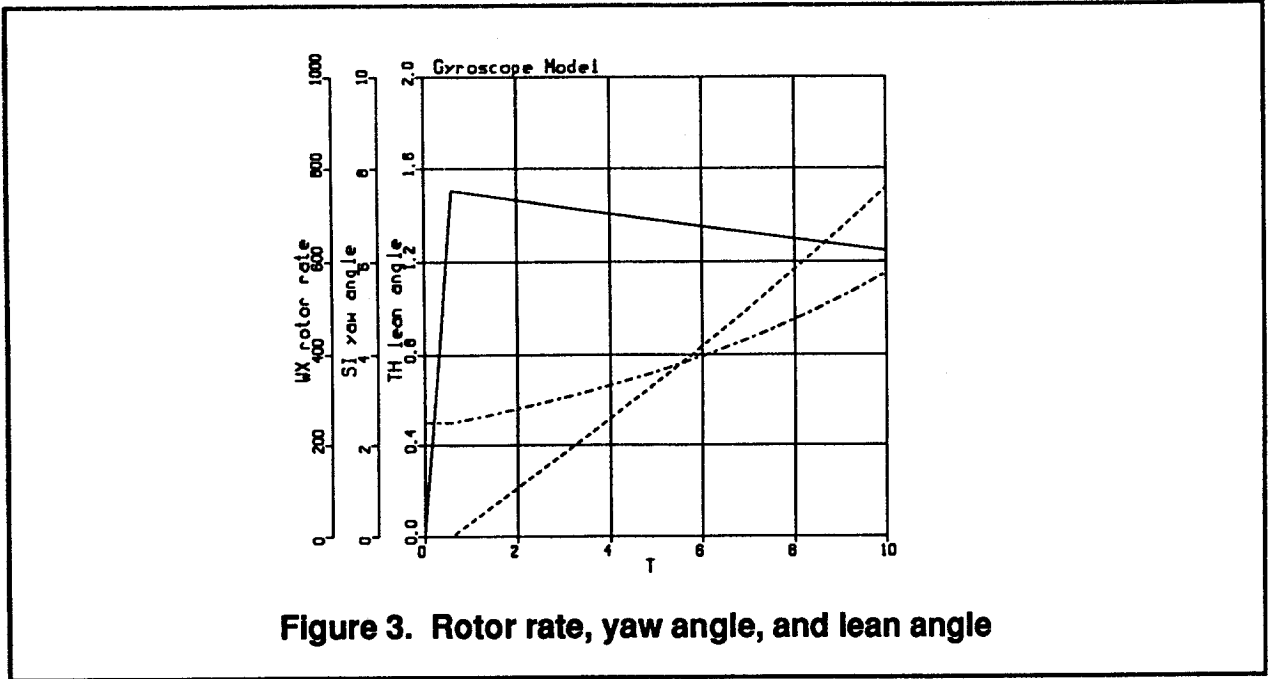


Figure 3. Rotor rate, yaw angle, and lean angle

Running the Model

When the model definition program has been written, ACSL is called. It translates and compiles the program, producing an executable simulation module. When this is executed, it reads and executes runtime commands. That is, it shows you the ACSL runtime prompt (ACSL>) and executes the commands you enter. You can change any of the conditions defined in CONSTANT statements and rerun the program, testing your theories.

Starting a run The following sequence of typical runtime commands specifies a title, requests that all variables (excluding constants) be recorded during runs, and then runs the model.

```
ACSL> SET TITLE='Gyroscope Model'
ACSL> PREPARE/ALL
ACSL> START
```

This sample run uses the initial conditions established in the model definition. It goes through the wind-up sequence and then ten seconds of precession.

Plot rate and angles The rotor rate (wx), yaw angle (si), and lean angle (th) are plotted in Figure 3 with the command:

```
ACSL> PLOT wx/TAG='rotor rate' &
         si/STYLE=3/TAG='yaw angle' &
         th/STYLE=5/TAG='lean angle'
```

The /STYLE qualifier specifies the dot/dash pattern of the lines. From the plot, you can see that the rotor speed increases during the string pull to about 750 rad/sec (7000 rpm) and then falls off over the ten second simulation period to just over 600 rad/sec. Lean angle starts off at 0.5 rad (about 30 degrees) and rises to about 1.2 rad while the gyro precesses over one revolution (7.5 rad).

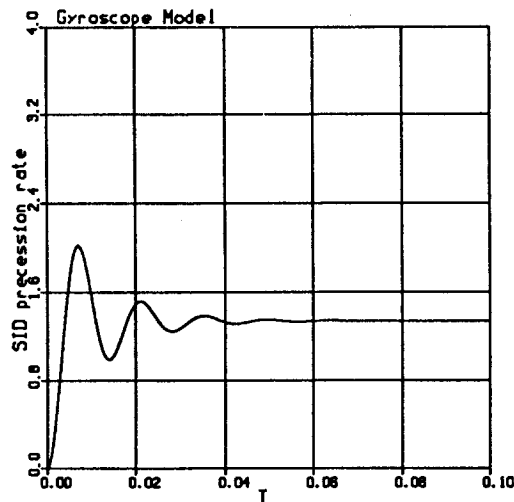


Figure 4. Yaw rate showing peak rate of 2 rad/sec

Experimenting with the model

One of the nice features of an ACSL simulation model is the ease with which changes can be made to the parameters and various experiments performed. It is interesting to look at the details of the nutation, which is the rapid oscillation during which the tip over from gravity is converted to a steady precession rate.

We first change the system so it starts up with a known rotor speed by skipping over the wind-up period. Setting the string length to a negative number skips the wind-up phase, and setting the initial rotor speed to 400 rad/sec establishes this as the initial condition.

```
ACSL> SET lstring=-10, wxz=400
```

The nutation frequency ($\omega_X I_X / I_Y$) is about 450 rad/sec. To see the details of the motion at this high frequency, we set the data logging rate higher (0.5 msec) and shorten the run time limit (to 100 msec). The command for this is:

```
ACSL> SET cint=0.0005, tstop=0.099
```

Figure 4 shows the plot of yaw rate (sid) as a function of time for these conditions which shows a peak rate of about 2 rad/sec but a steady state precession of about 1.5 rad/sec. The nutation oscillation with a period of about 0.013 msec can be clearly seen.

To further see the details of the nutation, we start the run with an initial angular rate of 30 rad/sec about the GY axis. This is equivalent to tapping the gyro in the direction of the lean angle. In order to slow down the damping of the oscillator, the rate damping gain K_{GF} is reduced by a factor of ten. The runtime commands for this experiment are:

```
ACSL> SET wyz=-30, kgf=0.0004, tstop=0.300
ACSL> START
```

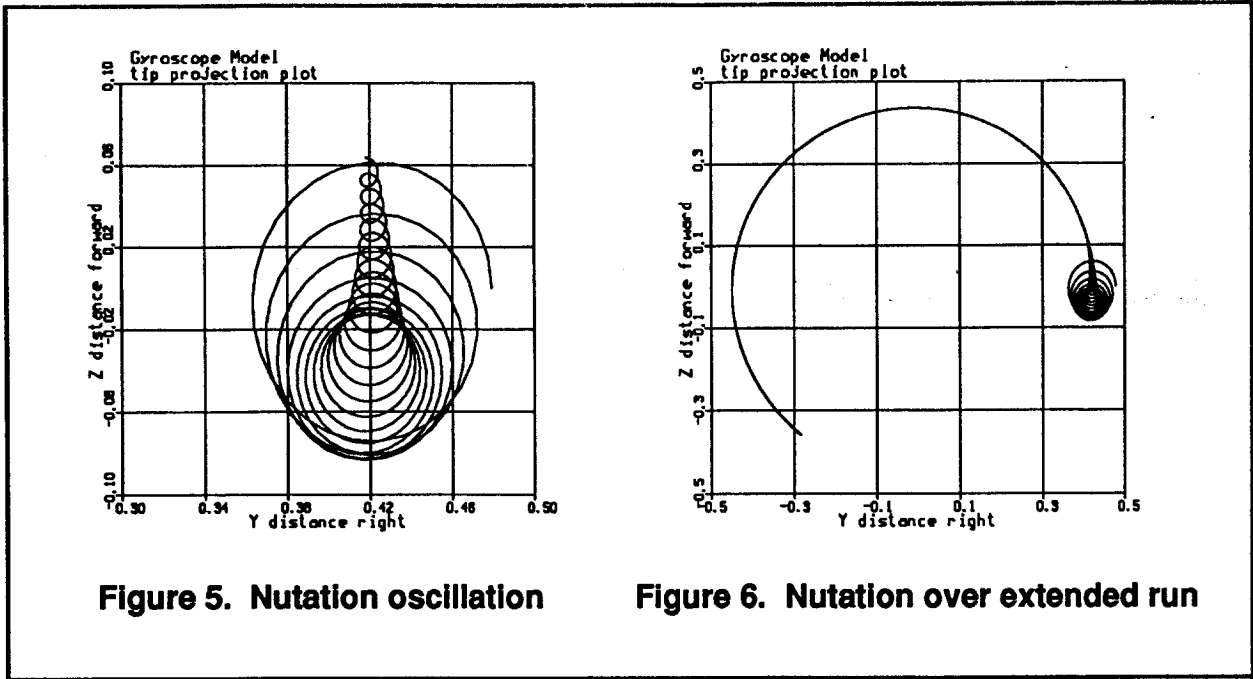


Figure 5. Nutation oscillation

Figure 6. Nutation over extended run

After this run, plot the y/z position as follows:

```
ACSL> PLOT/XAXIS=y z
```

Looking at the plot in Figure 5, the nutation oscillation is seen clearly because of the artificially low damping. We also see the motion changing to a precession to the left because of gravity.

Extending the run time shows the precession to the left turning into a circle. The following commands produce Figure 6:

```
ACSL> SET tstop=3
ACSL> START
ACSL> PLOT/XAXIS=y z
```