THE SYMMETRIC SPINNING TOP

DERIVING THE EQUATIONS OF MOTION

Spinning tops have to be visualised in reference to 2 frames - the space frame \((x, y, z)\) that is effectively the 'real life' frame that we have been using up until now when describing the position of the pendulums; and the body frame \((x', y', z')\) that is basically another set of \((x, y, z)\) axes taken with the \(z'\)-axis pointing up through the middle of the spinning top. This body frame obviously changes its orientation with respect to the space frame as the spinning top moves.

There are 3 degrees of freedom needed to describe the position of the spinning top - \(\theta\), which is the angle between the \(z\)-axis of the space frame and the \(z'\)-axis of the body frame; \(\varphi\), which is the rotation angle of the \(z'\)-axis of the body frame around the \(z\)-axis of the space frame; and \(\psi\), which is the rotation angle of the spinning top around its own \(z'\)-axis (that of the body frame).

![Figure 1: The Symmetric Spinning Top](image)

Note that \(\alpha\) is the angle between the \(z'\)-axis of the body frame and the outside of the spinning top, effectively determining the shape of the top.
When dealing with a spinning rigid body, moments of inertia must be considered. These influence the motion by taking into account the shape and density of the body, and thus translating them into the equations of motion.

There are 3 moments of inertia: $I_1$, $I_2$, and $I_3$. If the body is asymmetric then all 3 moments are different, if it is symmetric then 2 of the moments are equal, and if it is a sphere, then all 3 moments are equal.

Since the rigid body in question here is a cone, it is symmetric and thus $I_1 = I_2 \neq I_3$.

Scheck (1994) states that the moments of inertia for a symmetric top are:

$$I_1 = I_2 = \frac{3}{20} m \left( r^2 + \frac{l^2}{4} \right)$$

$$I_3 = \frac{3}{10} mr^2$$

Where $m$ is the mass of the top, $r$ is the radius of the top, and $l$ is the length of the top along the $z'$-axis from the origin (point of the top) to the centre of the circle at the other end of the top.

Note that although we are mainly interested in the space system (since this is what will be plotted in the applet), in order to find the necessary components for the space system we first need to find them in the body system and then apply translations.

José and Saletan (2002) show that the Lagrangian of the symmetric spinning top is:

$$L = \frac{1}{2} I_1 (\dot{\theta}^2 + \dot{\phi}^2 \sin^2(\theta)) + \frac{1}{2} I_3 (\dot{\psi} + \dot{\phi} \cos(\theta))^2 - mg l \cos(\theta)$$

Where $l$ is the length of the spinning top along the $z'$-axis from the origin (point of the top) to the centre of the circle at the other end of the top.

Using the property (1) from the documentation for the double pendulum, we know that:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta} = 0$$

(1)

Where:

$$\frac{\partial L}{\partial \theta} = I_1 \dot{\phi}^2 \sin(\theta) \cos(\theta) + \frac{\partial}{\partial \theta} \left[ \frac{1}{2} I_3 (\dot{\psi}^2 + 2 \dot{\phi} \dot{\psi} \cos(\theta) + \dot{\phi}^2 \cos^2(\theta)) \right] + mg l \sin(\theta)$$
\[ \frac{\partial L}{\partial \theta} = I_1 \phi^2 \sin(\theta) \cos(\theta) - I_3 \psi \phi \sin(\theta) - I_3 \phi^2 \cos(\theta) + mgl \sin(\theta) \]

\[ \frac{\partial L}{\partial \phi} = \phi^3 \sin(\theta) \cos(\theta) \left[ I_1 - I_3 \right] - I_3 \phi \psi \sin(\theta) + mgl \sin(\theta) \]

And:

\[ \frac{\partial L}{\partial \theta} = I_1 \dot{\theta} \]

\[ \frac{d}{dt} \left( \frac{\partial L}{\partial \theta} \right) = I_1 \ddot{\theta} \]

So equation (1) becomes:

\[ I_1 \dot{\theta} - \phi^2 \sin(\theta) \cos(\theta) \left[ I_1 - I_3 \right] + I_3 \phi \psi \sin(\theta) - mgl \sin(\theta) = 0 \]

So rearranging for \( \dot{\theta} \) we get:

\[ \dot{\theta} = \frac{\phi^2 \sin(\theta) \cos(\theta) \left[ I_1 - I_3 \right] - I_3 \phi \psi \sin(\theta) + mgl \sin(\theta)}{I_1} \]

(2)

Again, using the property (1) from the documentation for the double pendulum, we know that:

\[ \frac{d}{dt} \left( \frac{\partial L}{\partial \phi} \right) - \frac{\partial L}{\partial \phi} = 0 \]

(3)

Where:

\[ \frac{\partial L}{\partial \phi} = 0 \]

And:

\[ \frac{\partial L}{\partial \phi} = I_1 \phi \sin^2(\theta) + \frac{\partial}{\partial \phi} \left[ \frac{1}{2} I_3 (\psi^2 + 2 \phi \psi \cos(\theta) + \phi^2 \cos^2(\theta)) \right] \]

\[ \frac{\partial L}{\partial \phi} = I_1 \phi \sin^2(\theta) + I_3 \psi \cos(\theta) + I_3 \phi \cos^2(\theta) \]
So:
\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot\phi} \right) = I_1\dot\phi \sin^2(\theta) + 2I_1\dot\phi \dot\theta \sin(\theta) \cos(\theta) + I_3\dot\psi \cos(\theta) - I_3\dot\psi \dot\theta \sin(\theta) \\
-2I_3\dot\phi \dot\theta \sin(\theta) \cos(\theta) + I_3\dot\phi \cos^2(\theta)
\]

Therefore, equation (3) becomes:

\[
I_1\ddot\phi \sin^2(\theta) + 2I_1\ddot\phi \dot\theta \sin(\theta) \cos(\theta) + I_3\ddot\psi \cos(\theta) - I_3\ddot\psi \dot\theta \sin(\theta) - 2I_3\ddot\phi \dot\theta \sin(\theta) \cos(\theta) \\
+ I_3\dot\phi \cos^2(\theta) = 0
\]

Rearranging for \( \ddot\phi \) gives:

\[
\ddot\phi = \frac{2[I_3 - I_1]\ddot\phi \dot\theta \sin(\theta) \cos(\theta) - I_3\ddot\psi \cos(\theta) + I_3\ddot\psi \dot\theta \sin(\theta)}{I_1 \sin^2(\theta) + I_3 \cos^2(\theta)}
\]

(4)

Once more, by the property (1) from the documentation for the double pendulum we have:

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot\psi} \right) - \frac{\partial L}{\partial \psi} = 0
\]

(5)

Where:

\[
\frac{\partial L}{\partial \psi} = 0
\]

And:

\[
\frac{\partial L}{\partial \dot\psi} = \frac{\partial}{\partial \dot\psi} \left[ \frac{1}{2} I_3(\dot\psi^2 + 2\dot\phi \dot\psi \cos(\theta) + \dot\phi^2 \cos^2(\theta)) \right]
\]

\[
\frac{\partial L}{\partial \dot\psi} = I_3\dot\psi + I_3\dot\phi \cos(\theta)
\]

So:

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot\psi} \right) = I_3\ddot\psi + I_3\dot\phi \cos(\theta) - I_3\ddot\phi \dot\theta \sin(\theta)
\]
And equation (5) becomes:

\[ I_3 \ddot{\psi} + I_3 \dot{\phi} \cos(\varphi) - I_3 \dot{\phi} \dot{\theta} \sin(\theta) = 0 \]

Rearranging for \( \psi \) gives:

\[ \dot{\psi} = \frac{I_3 \dot{\phi} \dot{\theta} \sin(\theta) - I_3 \dot{\phi} \cos(\theta)}{I_3} \]

Simplifying this gives us:

\[ \dot{\psi} = \dot{\phi} \dot{\theta} \sin(\theta) - \dot{\phi} \cos(\theta) \quad (6) \]

It can be seen that equations (4) and (6) depend upon each other, and thus must be solved simultaneously to avoid recursion in the numerical integration stage.

Firstly solving for \( \dot{\phi} \), we can substitute the equation (6) into the equation (4) as follows:

\[ \dot{\phi} = \frac{2[I_3 - I_1] \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) - I_3 (\dot{\phi} \dot{\theta} \sin(\theta) - \dot{\phi} \cos(\theta)) \cos(\theta) + I_3 \psi \dot{\theta} \sin(\theta)}{I_1 \sin^2(\theta) + I_3 \cos^2(\theta)} \]

Now, expanding and rearranging we get:

\[ \dot{\phi} (I_1 \sin^2(\theta) + I_3 \cos^2(\theta)) = 2[I_3 - I_1] \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) - I_3 \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) + I_3 \dot{\phi} \dot{\theta} \sin(\theta) + I_3 \psi \dot{\theta} \sin(\theta) \]

Factorising \( \dot{\phi} \):

\[ \dot{\phi} \sin^2(\theta) = 2[I_3 - I_1] \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) - I_3 \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) + I_3 \psi \dot{\theta} \sin(\theta) \]

So finally:

\[ \dot{\phi} = \frac{2[I_3 - I_1] \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) - I_3 \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) + I_3 \psi \dot{\theta} \sin(\theta)}{I_1 \sin^2(\theta)} \quad (7) \]

Next solving for \( \psi \), we substitute equation (4) into equation (6):

\[ \psi = \dot{\phi} \dot{\theta} \sin(\theta) - \left[ \frac{2[I_3 - I_1] \dot{\phi} \dot{\theta} \sin(\theta) \cos(\theta) - I_3 \psi \cos(\theta) + I_3 \psi \dot{\theta} \sin(\theta)}{I_1 \sin^2(\theta) + I_3 \cos^2(\theta)} \right] \cos(\theta) \]
Expanding and rearranging we get:

\[
\frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\psi}}{\cos(\theta)} = \frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\phi}\dot{\sin(\theta)}}{\cos(\theta)} - 2[l_3 - l_1]\ddot{\phi}\dot{\theta}\dot{\sin(\theta)}
\]

Thus:

\[
\frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\psi}}{\cos(\theta)} = \frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\phi}\dot{\sin(\theta)}}{\cos(\theta)} - 2[l_3 - l_1]\ddot{\phi}\dot{\theta}\dot{\sin(\theta)} + l_3\dot{\psi}\dot{\cos(\theta)} - l_3\ddot{\psi}\dot{\sin(\theta)}
\]

Factorising \(\ddot{\psi}\):

\[
\ddot{\psi} \left[ \frac{l_1 \sin^2(\theta) + l_3 \cos^2(\theta)}{\cos(\theta)} - l_3 \cos(\theta) \right] = \frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\phi}\dot{\sin(\theta)}}{\cos(\theta)} - 2[l_3 - l_1]\ddot{\phi}\dot{\theta}\dot{\sin(\theta)} - l_3\ddot{\psi}\dot{\theta}\sin(\theta)
\]

So:

\[
\ddot{\psi} \left[ \frac{l_1 \sin^2(\theta)}{\cos(\theta)} \right] = \frac{(l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\phi}\dot{\sin(\theta)}}{\cos(\theta)} - 2[l_3 - l_1]\ddot{\phi}\dot{\theta}\dot{\sin(\theta)} - l_3\ddot{\psi}\dot{\theta}\sin(\theta)
\]

And finally:

\[
\ddot{\psi} = \frac{\cos(\theta) \left[ \frac{l_1 \sin^2(\theta) + l_3 \cos^2(\theta))\ddot{\phi}\dot{\sin(\theta)}}{\cos(\theta)} - 2[l_3 - l_1]\ddot{\phi}\dot{\theta}\dot{\sin(\theta)} - l_3\ddot{\psi}\dot{\theta}\sin(\theta) \right]}{l_1 \sin^2(\theta)}
\]

(Equation 8)

Equations (2), (7) and (8) are the only equations needed to describe the position of the top at time \(t\), since we can integrate them numerically (as before with the pendulums).

**PROGRAMMING THE JAVA APPLET**

It can be seen that a similar problem exists here as that which was encountered when programming the applet for the spherical pendulum. Both the equations for \(\ddot{\phi}\) and \(\ddot{\psi}\) have multiples of \(\sin(\theta)\) in the denominator. Therefore, when \(\theta = n\pi\), \(\ddot{\phi}\) and \(\ddot{\psi}\) tend to infinity and the system becomes unstable. However, as with the spherical pendulum it is possible to program an applet to show the motion when it is stable.
Unfortunately, we cannot merely use polar spherical coordinates to transform $\theta$, $\varphi$, and $\psi$ into $x$, $y$ and $z$ coordinates (the space frame). Thus I had to find a way to do this in order to be able to plot the spinning top in an applet.

To begin with, I set about devising a way to plot a reference image of the spinning top in the body frame ($x'$, $y'$, $z'$) that I could transform later.

![Figure 2: Image of a Symmetric Spinning Top](image)

It can be seen from Figure 2 that if we can find the points $q_i$, then the spinning top can be plotted by drawing lines from the origin $O$ to each point. The shape of the spinning top then appears, leaving only to connect the points $q_i$ together to form a ‘rim’.

The ($x'$, $y'$, $z'$) coordinates of each $q_i$ can be described as:

\[
\begin{align*}
  x' &= r' \cos(\beta) \\
  y' &= r' \sin(\beta) \\
  z' &= l'
\end{align*}
\]

Where the radius $r'$ of the image is given by:

\[r' = l' \tan(\alpha)\]

Note that $l'$ is the (constant) length from the origin $O$ to the centre of the circle at the other end of the top in pixels; $\alpha$ is a constant angle for all $q_i$ (I took it to be $\frac{\pi}{6}$ in the applet); and $\beta$ is the rotation about the $z'$ axis. This changes for each $q_i$, for example if at $q_0$, $\beta = 0$, then at $q_1$, $\beta = \frac{1\pi}{5}$ (since there are 10 lines used in Figure 2). For $n$ lines, $\beta = \frac{2i\pi}{n}$.
This provided the information needed to plot the points $q_i$, and therefore the information needed to plot a reference image of the spinning top in the body frame $(x', y', z')$. I stored the data for $x'$, $y'$, $z'$ and $\beta$ at each $q_i$ in separate arrays in order to perform the necessary calculations in ‘for’ loops to keep the Java code a sensible size.

To translate this image into the space frame $(x, y, z)$ coordinates that were required in order to plot it as an applet, I first needed a transformation matrix.

Whittaker (1927) states this transformation to be:

$$ A = \begin{bmatrix}
\cos(\phi)\cos(\theta)\cos(\psi) - \sin(\phi)\sin(\psi) & \sin(\phi)\cos(\theta)\cos(\psi) + \cos(\phi)\sin(\psi) & -\sin(\theta)\cos(\psi) \\
-\cos(\phi)\cos(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & -\sin(\phi)\cos(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \sin(\theta)\sin(\psi) \\
\cos(\phi)\sin(\theta) & -\sin(\phi)\sin(\theta) & \cos(\theta)
\end{bmatrix} $$

And McCauley (1997) asserts that:

$$ x' = Ax $$

Thus:

$$ x = A^{-1}x' $$

I used the mathematical computer package ‘Maple’ to calculate the inverse of the transformation matrix $A$, and found it to be:

$$ A^{-1} = \begin{bmatrix}
\cos(\phi)\cos(\theta)\cos(\psi) - \sin(\phi)\sin(\psi) & -\cos(\phi)\cos(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) & \cos(\phi)\sin(\theta) \\
\sin(\phi)\cos(\theta)\cos(\psi) + \cos(\phi)\sin(\psi) & -\sin(\phi)\cos(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & -\sin(\phi)\sin(\theta) \\
-\sin(\theta)\cos(\psi) & \sin(\theta)\sin(\psi) & \cos(\theta)
\end{bmatrix} $$

Next I created some basic Java code to perform the matrix-vector multiplication required to find the $(x, y, z)$ coordinates – I assigned each element of $A^{-1}$ to variables $a_{11}$, $a_{12}$, $a_{33}$, etc.; where $a_{bc}$ is the matrix element in row $b$, column $c$. The computer then performed the calculations and stored the results in 3 separate arrays for $x$, $y$ and $z$:

$$ x[i] = a_{11}x'[i] + a_{12}y'[i] + a_{13}z'[i] + x_{REF} $$

$$ y[i] = a_{21}x'[i] + a_{22}y'[i] + a_{23}z'[i] + y_{REF} $$

$$ z[i] = a_{31}x'[i] + a_{32}y'[i] + a_{33}z'[i] + z_{REF} $$

Note that $x[i]$, $y[i]$ and $z[i]$ represent the $i^{th}$ element in the arrays storing the $x$, $y$, and $z$ coordinates of the points $q_i$ of the real image of the top. Similar notation is used to denote the $i^{th}$ elements in the arrays storing the $x'$, $y'$ and $z'$ coordinates of the points $q_i$ of the
$x_{\text{REF}}$, $y_{\text{REF}}$, and $z_{\text{REF}}$ are the coordinates on the applet for the origin $O$. Since I only intended to plot $x$ against $z$, and have the $y$-axis coming out of the computer screen towards the user (as with the spherical pendulum), I took the $y_{\text{REF}}$ value to be zero in preparation to make the ‘perspective’ calculations more straightforward.

I plotted lines between each of the $(x[i], y[i], z[i])$ coordinates and the origin $O$ in order to form a basic image of the top at time $t$. I then plotted lines between each point $q_i$ to form a ‘rim’ for the top.

At this point I had a perfectly formed top spinning on the screen, but (as with the spherical pendulum) I was having trouble visualising whether it was pointed towards or away from me. Thus I devised a way of creating the illusion of perspective.

To begin with, I needed to decide exactly how big the top would appear both when it was closest to the user (at $y_{\text{MAX}} = l$, where $y$ is the position of the point in the centre of the ‘wide’ end of the spinning top along the $y$-axis) and when it was furthest away from the user (at $y_{\text{MIN}} = -l$). Since the angle $\alpha$ governed the size of the top, it seemed prudent to use 2 values close to my (previously constant) $\alpha$ value. I decided the top should have $\alpha_{\text{MAX}} = \frac{\pi}{4}$ when $y = y_{\text{MAX}} = l$ and $\alpha_{\text{MIN}} = \frac{\pi}{8}$ when $y = y_{\text{MIN}} = -l$.

I then introduced a new variable:

$$Y = y_{\text{MAX}} + y$$  \hspace{1cm} (9)

Remember $y$ can be positive or negative.

So:

$$Y_{\text{MAX}} = 2y_{\text{MAX}}$$
$$Y_{\text{MIN}} = 0$$  \hspace{1cm} (10)

Using $Y$ meant the perspective calculations could be performed using only positive numbers, since $Y \geq 0$. 
By studying Figure 3, it can be seen that:

\[ \alpha = \mu Y + \alpha_{MIN} \]

Where:

\[ \mu = \frac{\alpha_{MAX} - \alpha_{MIN}}{Y_{MAX} - Y_{MIN}} \]

And since \( Y_{MIN} = 0 \):

\[ \mu = \frac{\alpha_{MAX} - \alpha_{MIN}}{Y_{MAX}} \]

Therefore:

\[ \alpha = \left( \frac{\alpha_{MAX} - \alpha_{MIN}}{Y_{MAX}} \right) Y + \alpha_{MIN} \quad (11) \]

And since we are using \( y \) as the position along the \( y \)-axis of the point in the centre of the ‘wide’ end of the spinning top, we can use polar spherical coordinates to find this value as follows:

\[ y = l' \sin(\theta) \cos(\varphi) \]

We can clearly see from this that \( y_{MAX} = l' \) at \( \sin(\theta) = \cos(\varphi) = 1 \). Substituting this into equation (9) gives:

\[ Y = l' + l' \sin(\theta) \cos(\varphi) \]
And equation (10) becomes:

\[ Y_{MAX} = 2l' \]

Substituting this information into equation (11) we get:

\[ \alpha = \left( \frac{\alpha_{MAX} - \alpha_{MIN}}{2l'} \right) (l' + l' \sin(\theta) \cos(\phi)) + \alpha_{MIN} \]

Simplifying:

\[ \alpha = \left( \frac{\alpha_{MAX} - \alpha_{MIN}}{2} \right) (1 + \sin(\theta) \cos(\phi)) + \alpha_{MIN} \]

And remember I chose \( \alpha_{MAX} = \frac{\pi}{4} \) and \( \alpha_{MIN} = \frac{\pi}{8} \), so the formula I used in my Java code was:

\[ \alpha = \left( \frac{\pi}{4} - \frac{\pi}{8} \right) (1 + \sin(\theta) \cos(\phi)) + \frac{\pi}{8} \]

Unfortunately, even with the size of the top dynamically changing with \( y \), I still found it difficult to tell whether it was facing towards or away from me when \( |y| \) was small. Thus I set about creating a ‘lid’ for the top – then the ‘body’ of the spinning top would appear over the lid when it was facing away from the user, and the lid would appear over the body when it was facing towards them.

I drew the lid in the same ‘for’ loop as the rest of the spinning top – using 2 lines of code very similar to that which I used to draw the ‘rim’ of the top.

In order to draw the rim, I drew lines between the \((x[i], z[i])\) coordinates and the \((x[(i+1)\%n], z[(i+1)\%n])\) coordinates (where \% signifies ‘modulo’, \( i \) is the array address of the point \( q_i \), and \( n \) is the total number of points \( q \) there are to plot). The modulo operation is needed to make the arrays ‘circular’ so they can travel easily between the first and last elements (i.e. if there are 10 points, then when the ‘for’ loop asks to plot a line between \((x[9], z[9])\) and \((x[10], z[10])\) then the \%n tells the computer to actually draw a line between \((x[9], z[9])\) and \((x[0], z[0])\); hence saving the computer from drawing to \((x[10], z[10]),\) which does not exist).

Using this same structure, I introduced 2 new sets of lines to be plotted:

1. Lines from \((x[i], z[i])\) to \((x[(i+\frac{\pi}{4})\%n], z[(i+\frac{\pi}{4})\%n])\)
2. Lines from \((x[i], z[i])\) to \((x[(i+\frac{\pi}{2})\%n], z[(i+\frac{\pi}{2})\%n])\)
The first set of lines was plotted between every point $q_i$ and the point directly opposite it $q_{i+n}$. The second set of lines was plotted between every point $q_i$ and the point a third of the way around the top away from it $q_{i+n/3}$ (I used 18 points so my $n$ was divisible by both 2 and 3).

The 3 sets of lines (including the rim) together created a spirograph effect as shown in **Figure 4**.

![Figure 4: The Spinning Top Lid](image)

Now the lid was created, it remained to find a way to plot it over or underneath the body of the spinning top, depending on whether it was facing towards or away from the user.

In order to find which way the top was facing, I used a loop to find which point $q_i$ had the maximum value of $y$, and store this maximum in a temporary variable (say $\kappa$). I then compared this value to that of the radius $r'=l\tan(\alpha)$. If $\kappa = r'$ then the spinning top is exactly vertical, and so is neither facing towards nor away from the user. By this reasoning, if $\kappa > r'$ then the top is facing towards the user, and if $\kappa < r'$ then the top is facing away from the user. Using these results, I plotted the lid after the body when the top was facing towards the user (so the lid appeared on top of the body), and plotted the lid before the body when the top was facing away from the user (so the lid appeared underneath the body).

This helped slightly, but not as much as I had hoped. I then tried colour-coding the spinning top. I plotted the body in one colour, the top (when facing towards the user) in another, and the top (when facing away from the user) in another. I found that using bright colours for the body and darker colours for the lid worked best, and when I plotted the lid ‘facing away’ in a grey colour it looked like you were looking at the underneath of the lid. This made the top much easier to visualise, and I made it clearer still by changing the background colour to black (as opposed to white) – this made everything stand out more.

With this done, there was one final problem relating to the visual aspects of the top, and this was seeing how fast the pendulum was spinning. I experimented with the frame-rate of the animation, but obviously if I altered this it would mean the top would not be spinning in real time. Instead, I opted for plotting a small red circle over one of the points (say $q_0$). It was then easy to see this point moving around in respect to the rest of the spinning top and so
gauge how fast it was spinning. I made this dot disappear when the top was facing away from
the user so it appeared to be on top of the lid.

I now considered my applet effectively finished, and so added text fields for the user to input
their own values for $m$, $\varphi$, $\psi$ etc. I also added a checkbox that activated and deactivated
gravity (by switching the value for $g$ between 0 and 9.8); a checkbox that enabled the user
to activate and deactivate the so-called ‘reference dot’ (that helps visualise how fast the top is
spinning); a button so the user can start or stop the applet as they like; and finally a scroll bar
that allows the user to adjust the frame-rate of the animation as they wish. I thought this
would be useful in that the user could slow the animation if they wished to look at the motion
in more detail. Obviously, since I added this feature it was prudent to print the time of the
animation (in seconds) onto the applet, so that the user still had a sense of how fast the top
was spinning even if they had adjusted the frame-rate.