CAT FLIPPING

ARDEN, SEAN, AND WILL

1. Introduction

In this document we explore the mathematical framework surrounding our cat model, provide details about the physics simulation we used, and recount mathematical formulations used in our code. We begin ,in the first section, by proving that our axis of rotation is a principle axis, then move on to calculate the moment of inertia and verify that our model truly does have zero angular momentum. In the next section we detail our code, giving a run down of the simulation. And the final section includes how we smoothed our angular velocity function and a computation solving for the initial angle needed for our model to perform a complete rotation.

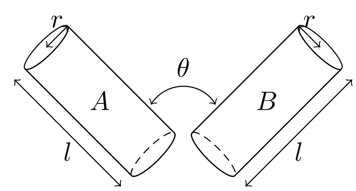

Figure 1. The model of a cat, and the variable parameters.

2. Moment of Inertia of the Cat

We model a falling cat with two cylinders A and B connected by a spherical joint (which we assume to be massless) with A and B separated by angle θ.

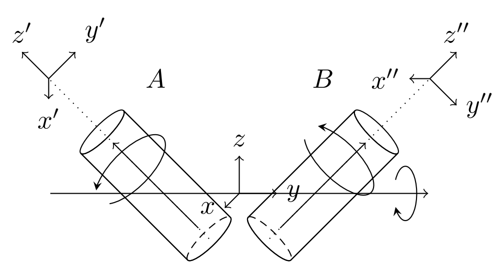

Figure 2. The coordinate axis that we define for our computations.

Claim 1. About the center of mass, the y axis of the cat is principle. Further, we can write $\vec{L} = (2I_{yy})\omega_y\hat{y}$. For an I_{yy} dependent on the geometry of the cylinders and θ.

Proof. Let the moment of inertia tensors of cylinders A and B about the center of mass of the cat as a whole be denoted I_A and I_B. Recall the definition of the moment of inertia tensor:

$$I = \begin{pmatrix} \sum(y^2+z^2) & \sum xy & \sum xz \\ \sum yx & \sum(x^2+z^2) & \sum yz \\ \sum zx & \sum zy & \sum(x^2+y^2) \end{pmatrix}$$

Then, by the reflectional symmetry of bodies A and B, the calculations for I_A and I_B are identical except for the replacement of y by $-y$. So, we can safely write

$$I_A = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix} \text{ and } I_B = \begin{pmatrix} I_{xx} & -I_{xy} & I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ I_{zx} & -I_{zy} & I_{zz} \end{pmatrix},$$

And so the moment of inertia tensor I of the cat as a whole is of the form

$$I = I_A + I_B = \begin{pmatrix} 2I_{xx} & 0 & 2I_{xz} \\ 0 & 2I_{yy} & 0 \\ 2I_{zx} & 0 & 2I_{zz} \end{pmatrix}$$

Then $\vec{y}$ is a clear eigenvector of I confirming that the y axis is a principle axis. □

Note that the above argument can be generalized to any two bodies with reflectional symmetry through the x, y plane.

Claim 2. The I_{yy} in claim 2 can be written $I_{y'y'}\cos^2(\theta/2) + I_{z'z'}\sin^2(\theta/2)$ where $I_{y'y'}$ and $I_{z'z'}$ are the moment of inertia elements corresponding to the principle axes.

Proof. Let x', y', z' denote the principle axes of cylinder A about the center of mass with z' going down the length of the cylinder, x' parallel to x, and y' in the corresponding location for a right-handed orthogonal coordinate system. Then, with this coordinate system corresponding to the principle axes, we can write the moment of inertia tenor I'_A of cylinder A about the center of mass of A.

$$I_A = \begin{pmatrix} I_{x'x'} & 0 & 0 \\ 0 & I_{y'y'} & 0 \\ 0 & 0 & I_{z'z'} \end{pmatrix}$$

Now, we adjust to the x, y, z coordinate system established above and solve for I_{yy}. Note $\hat{y}$ is given by $(0, \cos(\theta/2), \sin(\theta/2))$ in x', y', z' coordinates. Thus by adjusting coordinates we have

$$\begin{aligned} I_{yy} &= \begin{pmatrix} 0 & \cos(\theta/2) & \sin(\theta/2) \end{pmatrix} \begin{pmatrix} I_{x'x'} & 0 & 0 \\ 0 & I_{y'y'} & 0 \\ 0 & 0 & I_{z'z'} \end{pmatrix} \begin{pmatrix} 0 \\ \cos(\theta/2) \\ \sin(\theta/2) \end{pmatrix} \\ &= I_{y'y'}\cos^2(\theta/2) + I_{z'z'}\sin^2(\theta/2) \end{aligned}$$

The above gives the value of I_{yy} about the center of mass of A, but the center of mass of the cat as a whole shares identical x and z values. So, by $I_{yy} = \sum x^2 + z^2$, the value of I_{yy} holds about the center of mass the cat.

□

Again, note the above argument is generalizable given the principle axes.

Then overall we have a simple relationship between the angular momentum of the cat and an angular velocity ω of the cat in the $\vec{y}$ direction.

$$\vec{L} = (I_{y'y'} \cos^2(\theta/2) + I_{z'z'} \sin^2(\theta/2))\omega\hat{y} \tag{1}$$

In particular, if A and B are two solid cylinders of equal density with length L, radius R, and mass M we have

$$\vec{L} = \left(\left(\frac{1}{4}MR^2 + \frac{1}{12}ML^2\right)\cos^2(\theta/2) + \frac{1}{2}MR^2\sin^2(\theta/2)\right)\omega\hat{y} \tag{2}$$

3. Note on stability

The cat rotates about y axis, but if this axis is unstable, the cat with have difficulty performing this turn. So, we must check the conditions under which the rotation axis is stable. Note that we have the following principle axis values (found in the same way we get I_{yy} above).

$$\begin{aligned}
I_{xx} &= I_{x'x'} \\
I_{yy} &= I_{y'y'} \cos^2(\theta/2) + I_{z'z'} \sin^2(\theta/2) \\
I_{zz} &= I_{y'y'} \sin^2(\theta/2) + I_{z'z'} \cos^2(\theta/2)
\end{aligned}$$

Assuming the cylindrical model and a sufficiently large length we can assume $I_{x'x'} = I_{y'y'} > I_{z'z'}$. But then, we have that

$$I_{yy} = I_{y'y'} \cos^2(\theta/2) + I_{z'z'} \sin^2(\theta/2) < I_{x'x'} \cos^2(\theta/2) + I_{x'x'} \sin^2(\theta/2) = I_{x'x'} = I_{xx}.$$

So, $I_{yy} > I_{xx}$ and by the same reasoning, $I_{zz} > I_{xx}$. Additionally, if we have $\pi \le \theta \le \pi/2$, then by the combination of $\sin(\theta/2) \le \cos(\theta/2)$, $I_{z'z'} < I_{y'y'}$, and the rearrangment inequality to conclude that

$$I_{yy} = I_{y'y'} \cos^2(\theta/2) + I_{z'z'} \sin^2(\theta/2) \le I_{y'y'} \sin^2(\theta/2) + I_{z'z'} \cos^2(\theta/2) = I_{zz}$$

And so we conclude that with a sufficiently large length and with $\pi \le \theta \le \pi/2$ we have that $I_{yy} \le I_{zz} < I_{xx}$, making I_{yy} not the intermediate axis and thus a stable axis.

Specificially, a "sufficiently large length" is exactly when $L \ge \sqrt{3}R$ for then assuming cylinders of length L, radius R, and mass M we get that

$$\begin{aligned}
I_{z'z'} &= \frac{1}{2}MR^2 = \frac{1}{4}MR^2 + \frac{1}{2}MR^2 \le \frac{1}{4}MR^2 + \frac{1}{2}M\left(\frac{L}{\sqrt{3}}\right)^2 \\
&= \frac{1}{4}MR^2 + \frac{1}{12}ML^2 = I_{x'x'}
\end{aligned}$$

which gives the assumption given earlier. So, both setting $\theta < \pi/2$ and $L < \sqrt{3}R$ could lead to issues with stability.

4. The Angular Velocity of Cylinders

Given the result of our minor project paper,

$$\text{Let } \mathcal{C} = 1 - 2\frac{I_{zz}}{I_{cc}}\sin(\theta/2)$$

While we solved for ω_0 as a specific value, we have the freedom for ω_0 to vary as a function of time. In general, we can have the function $\omega(t)$. Note that $\mathcal{C}$ is constant in time, so we only require that

$$\pi = \int \frac{d\phi}{dt}dt = \mathcal{C}\int \omega(t)dt$$

In other words, we can have any function $\omega(t)$ so long as

$$\int \omega(t)dt = \frac{\pi}{\mathcal{C}} = \omega_0 T$$

where T is the total time taken in the reorientation process.

We take of advantage of the previously described freedom over $\omega(t)$ to ensure continuity of angular velocity and angular acceleration throughout the reorientation maneuver. In particular for the purpose of the simulation we choose the function

$$\omega(t) = \omega_0\left(1 - \cos\left(\frac{\pi}{T}t\right)\right)$$

Note that upon integration we get $\int \omega(t)dt = \omega_0 T$ as required.

5. Angular Momentum Calculation

We will now demonstrate that our model has a net of zero angular momentum. Refer to the figure above. To compute the overall angular momentum of the body we will begin by calculating the angular momentum of the two respective cylinders. Let x', y', z' x'', y'' and z'' be the axes shown above 2.

In finding $\vec{L}'_A$ and $\vec{L}''_B$, we begin by noting that z' is a principle axis for cylinder A and z'' is a principle axis for cylinder B. To see why this is true remember that the moment of inertia tensor for a cylinder of height h and radius r about its central axis is

$$I = \begin{bmatrix} \frac{1}{12}m(3r^2+h^2) & 0 & 0 \\ 0 & \frac{1}{12}m(3r^2+h^2) & 0 \\ 0 & 0 & \frac{1}{2}mr^2 \end{bmatrix}.$$

The matrix above is diagonal so the central axis is a principal axis. Therefore rotation about the central axis ω, direction indicated in the figure 2, is given by

$$\vec{L_A}' = \begin{bmatrix} 0 \\ 0 \\ -I_{z'z'}\omega \end{bmatrix} \quad \vec{L_B}'' = \begin{bmatrix} 0 \\ 0 \\ I_{z''z''}\omega \end{bmatrix}.$$

Next, we move on to calculate the angular momentum of the two cylinders viewed as a single body rotating about the y axis, which we will denote $\vec{L}_C$. In calculating $\vec{L}_C$ we first remember that y is a a principal axis for rotation of the two cylinders viewed as a single body. With this in mind, angular momentum for rotation about the y axis with speed ω_c, direction indicated in the figure 2, has the form

$$\vec{L_C} = \begin{bmatrix} 0 \\ I_{yy}\omega_c \\ 0 \end{bmatrix}$$

$\vec{L_A}''$, $\vec{L''_B}$, and $\vec{L_C}$ are expressed in different bases, our next step is to express $\vec{L_A}$ and $\vec{L_B}$ in terms of the x, y, z basis. Recall that the matrix for negative and positive rotation by $\frac{\theta}{2}$ about the x axis, denoted $R^x_{-\theta/2}$ and $R^x_{\theta/2}$ respectively are given by

$$R^x_{-\theta/2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\frac{\theta}{2}) & \sin(\frac{\theta}{2}) \\ 0 & -\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix} R^x_{\theta/2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ 0 & \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}.$$

Using the matrices above we can compute $\vec{L}_A(\vec{L}'_A$ in the x, y, z basis) and $\vec{L}_B(\vec{L}''_B$ in the x, y, z basis). Along with the information that $I_{zz} = I_{z'z'} = I_{z''z''}$, we have

$$\vec{L_A} = \begin{bmatrix} 0 \\ \sin(\theta/2) I_{zz}\omega \\ \text{-}\cos(\theta/2) I_{zz}\omega \end{bmatrix} \quad \vec{L_B} = \begin{bmatrix} 0 \\ \sin(\theta/2) I_{zz}\omega \\ \cos(\theta/2) I_{zz}\omega\omega \end{bmatrix}.$$

Now we are ready to compute the overall angular momentum of the system. To do so first recall the main outcome of our minor project,

$$\omega_c = \omega \left(-2 \frac{I_{zz}}{I_{yy}} \sin(\theta/2) \right).$$

Computing $\vec{L}_A + \vec{L}_B + \vec{L}_C$ we obtain

$$\begin{bmatrix} 0 \\ \sin(\theta/2) I_{zz}\omega \\ \text{-}\cos(\theta/2) I_{zz}\omega \end{bmatrix} + \begin{bmatrix} 0 \\ \sin(\theta/2) I_{zz}\omega \\ \cos(\theta/2) I_{zz}\omega\omega \end{bmatrix} + \begin{bmatrix} 0 \\ \omega\left(-2 I_{zz} \sin(\theta/2)\right) \\ 0 \end{bmatrix} = \vec{0}.$$

Our model has zero angular momentum.

6. Simulation

To verify our computations, we constructed a software simulation, that would be easily configurable, to allow rapid development. For the simulation we chose to use the `Python` programming language, and to make use of the `pyBullet` physics simulator. `pyBullet` is provided using python's integrated package manager `pip`.

6.1. **URDF File Format.** `pyBullet` makes use of the `URDF` file format, so all models in the simulation must be loaded from a urdf file. This format has some specific constraints that are explained here, and the cat model that we used is included. The urdf file type is a subset of XML.

The model is constructed with `links` and `joints`, and the entire format is contained within a `robot` object.

6.1.1. *Link.* The link object can contains three sub objects, `visual collision`, and `inertial`. The visual and collision objects, can define a `geometry`, which can in turn be constructed from a few selected primitives. And they can also define an `origin` object. The geometry is used to determine the visual display, or the collision geometry accordingly, and the origin object defines the orientation and the position of the link relative to the robots origin.

The `inertial` object can define the components mass, and its inertial tensor, along with the origin for the inertial computation. For most purposes the origins for all three of these components can be the same, and the geometry of the visual and collision components can also be the same.

6.1.2. *Joint.* The joint object defines the connections between the different links. Each joint has a parent link and a child link, and an origin. The types of joints that are available are a fixed joint, where no motion is permitted, a revolute joint, which is rotation about a single axis, and a slide joint, which is an in and out motion. Each of these different types of joints have different parameters that need to be defined for each of them. Our model only makes use of the fixed joints, and the revolute joints.

For the revolute joint, we must also define the axis for its rotation and the limits of angles for the rotation.

6.1.3. *Construction.* To construct the cat with variable parameters, we wrote a script to generate the `urdf` file altering the parameters to be what we desired.

6.2. **Simulation Setup.** `pyBullet` provides all of the physics simulations that are necessary for most common applications, but first it must be initialized in the code. To do this one must use the following commands before any physics simulation is possible.

```
import pybullet as p
physicsClient = p.connect(p.GUI)
p.setAdditionalSearchPath(pybullet_data.getDataPath())
```

These commands initialize the graphical interface for the simulation, and includes a path for some standard `pyBullet` models.

The next step is to set global constants that are persistent for the entirety of the simulation. In this case we will only set the gravitational constant.

```
p.setGravity(0,0,-9.8)
```

This function sets the gravitational force in the x, y, and z directions.

The final step before beginning the simulation is to load any models that will be used in the simulation. For our purposes, we load the standard `plane.urdf`, and we load our constructed `cat.urdf`.

```
planeId = p.loadURDF("plane.urdf")
startPos = [0,0,10]
startOrientation = p.getQuaternionFromEuler([0,0,0])
catId = p.loadURDF("cat.urdf",startPos, startOrientation)
```

These functions load the plane model positioned at the origin, and loads the cat model positioned at $(0, 0, 10)$, with the default orientation.

6.3. **Simulation Computation.** With our our organization of the simulation, we are able to compute all necessary values prior to the execution of the simulation. This process is implementing the equations that we found previously, and computing their numerical values for the provided model of the cat. The first stage is computing the time that the cat has to preform the maneuver, and partitioning that time for the three different stages of the motion.

```
t_max = np.sqrt(2 * 10 / 9.8)
step_2 = 2 * t_max / 10
step_3 = 8 * t_max / 10
dt = (step_3 - step_2)
```

Next we compute the moment of inertia.

```
ix = 1/12 * mass * (radius**2 + length**2)
iy = 1/12 * mass * (radius**2 + length**2)
iy = 1/2 * mass * radius**2
ic = iy * np.cos(theta / 2.0)**2 + iz * np.sin(theta / 2.0)**2
c = 1 - 2 * iz / (2 * ic) * np.sin(theta / 2.0)
```

These first lines are computing the moment of inertia in the three principal axes for each individual cylinder of the cat model. Next we compute the combined moment of inertia, and a constant that we call c. This constant c we derived previously. Now using the moment of inertia, and the time available for the cat, we compute the rate at which the cat should rotate. Since this will be variable with respect to time, we also construct a function to return the rate of rotation.

```
omega_0 = np.pi / (dt * cloud)
omega = lambda t: omega_0 * (-np.cos(np.pi / dt * t) + 1)
```

6.4. **Simulation Execution.** Now with all the components of the simulation constructed, we are able to run the simulation. The entire code is presented below, and we will explain the three stages of execution afterwards.

```
while i < 1000:
p.stepSimulation()
if t < step_2:
p.setJointMotorControlArray(catId, [0,2], p.POSITION_CONTROL,
targetPositions=[(np.pi - theta) / 2.0 * (1 - np.cos(np.pi / step_2 *
(t - step_2))),0])
elif t < step_3:
p.setJointMotorControlArray(catId, [0,2], p.POSITION_CONTROL,
targetPositions[(np.pi - theta) * np.cos(phi), (np.pi - theta) * np.sin(phi)])
phi += omega(t - step_2) / fps
elif t < t_max:
p.setJointMotorControlArray(catId, [0,2], p.POSITION_CONTROL,
targetPositions=[x/2 * (1 + np.cos(np.pi / dt_3 * (t - step_3))), y/2 * (1
+ np.cos(np.pi / dt_3 * (t - step_3)))])
t += 1.0 / fps
i += 1
```

The simulation can be broken into three steps. The first is the process of bending the straight cat into the partially bent position. Then the next stage is the rotational motion, which results in the reorientation. Then the final stage is the straightening of the cat.

7. Numerical Example

Here we will consider a physical example using our model of a cat, to compute all of the necessary values. We will consider a cat of `mass` $= m = 1$, `length` $= l = 1$, radius `radius` $= r = 0.1$, and maximum angle of bend $\theta = \frac{\pi}{4}$. We will also consider the cat to be dropped from a height of 10, and with standard gravity of -9.8. The model of this cat is depicted in Figure 3.

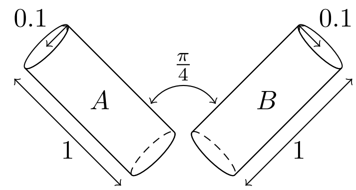

FIGURE 3. Specific model of cat for example

The first step is to compute the time that the cat has to do the rotation. We find this simply using basic kinematics

$$\Delta t = \sqrt{\frac{2h}{g}} = \sqrt{\frac{2 \cdot 10}{9.8}} \approx 1.428.$$

Next we will also compute the moment of inertia I, and to compute this we require the inertia tensor for each of the cylinders individually. So we compute these to be

$$\begin{pmatrix} \frac{m}{12}\left(r^2+l^2\right) & 0 & 0 \\ 0 & \frac{m}{12}\left(r^2+l^2\right) & 0 \\ 0 & 0 & \frac{mr^2}{2} \end{pmatrix} \approx \begin{pmatrix} 0.084 & 0 & 0 \\ 0 & 0.084 & 0 \\ 0 & 0 & 0.005 \end{pmatrix}.$$

Since both A and B cylinders are identical, then this is the inertia tensor for both of them. The next stage is to compute the moment of inertia in the $\vec{y}$ axis.

$$I_{yy} = I_{y'y'} \cos^2\left(\frac{\theta}{2}\right) + I_{z'z'} \sin^2\left(\frac{\theta}{2}\right) \approx 0.071841 + 0.000732 \approx 0.0726.$$

Then from our previous computation, we evaluate for the rotational speed of the cylinders ω. We compute this as

$$\omega = \frac{\pi}{\Delta t\left(1 - \frac{I_{z'z'}}{I_{yy}} \sin\left(\frac{\theta}{2}\right)\right)} \approx 2.2514.$$

This is the rate that each cylinder must rotate at in order for the full rotation to occurs.

To verify that this rate will work, we compute

$$\begin{aligned} &\int_0^{\Delta t} \omega \left(1 - \frac{I_{z'z'}}{I_{yy}} \sin\left(\frac{\theta}{2}\right)\right) dt \\ &= \frac{\pi}{\Delta t} \int_0^{\Delta t} dt = \pi. \end{aligned}$$

Now we can compute the maximum amount of torque that the cat would require to preform the rotation motion. This is given by

$$\max(\tau_2) = \omega \frac{\pi}{\Delta t} I_*.$$

We compute I_* by the use of the parallel axis theorem. So we find

$$I_* = I_{x'x'} + md = \frac{m}{12}\left(r^2+l^2\right) + m\frac{h^2}{4} = \frac{m}{12}\left(r^2+4l^2\right) \approx= 0.3342.$$

This is the moment of rotation of the cylinder about its end. Now we can use this to compute the maximum torque required.

$$\max(\tau_2) = \omega \frac{\pi}{\Delta t} I_* = \frac{\pi^2 I_*}{\Delta t^2 \left(1 - \frac{I_{z'z'}}{I_{yy}} \sin\left(\frac{\theta}{2}\right)\right)} \approx 1.655N.$$

8. Smoothing in the Simulation

For each of the three steps in the simulation, we apply a smooth curve to the function $\theta(t)$ to ensure continuity of angular velocity and angular acceleratioin, for the motors in the simulation can only handle a finite torque. Assume the model bends to angle θ_0 and we allow each of the three stages times T_1, T_2, and T_3 corresponding to the folding, the twisting, and the intwisting respectively. Then, we control the in the simulation according to functions θ_1, ω_2, θ_3 for each of the respective phases.

$$\theta_1(t) = \left(\pi - \frac{\theta_0}{2}\right)\left(1 - \cos\left(\frac{\pi}{T_1} t\right)\right)$$
$$\omega_2(t) = \omega_0 \left(1 - \cos\left(\frac{\pi}{T_2} t\right)\right)$$
$$\theta_3(t) = \left(\pi - \frac{\theta_0}{2}\right)\left(1 + \cos\left(\frac{\pi}{T_3} t\right)\right)$$

Additionally, the simulation software requires a provided maximum torque for the motors. A torque too low will prevent the cat from turning and we found a huge torque could be at the cost of precision in the simulation. So, we provide a calculation for placing reasonable bounds on the torque. Let τ_1, τ_2, τ_3 denote the torques for each of the three stages. Then each function is given by the following where I_* is the moment of inertia of the cylinder lengthwise about the end of the cylinder. Note this can be calculated by the parallel axis theorem with $I_* = I_{x'x'} + md^2$.

$$\tau_1 = I_* \ddot{\theta}_1 \qquad \tau_2 = I_* \dot{\omega}_2 \qquad \tau_3 = I_* \ddot{\theta}_3$$

We solve for each function and apply $\sin(t) \leq 1$ and $\cos(t) \leq 1$, which gives the maximum of the functions over the appropriate interval, so:

$$\max(\tau_1) = \frac{\pi^2}{T_1^2} I_* \qquad \max(\tau_2) = \omega_0 \frac{\pi}{T_2} I_* \qquad \max(\tau_3) = \frac{\pi^2}{T_3^2} I_*$$

providing the values to set the maximum torque to in each step.

9. Solving for one-turn θ

We solve for the θ between the cylinders such that the cat twists for a full 2π radians to accomplish a change in orientation of π radians. To accomplish this, we more generally consider when $\frac{d\phi}{dt} = \omega_0 \alpha$ for some $\alpha \in (0, 1)$ and notice that our original question corresponds to the case $\theta = \frac{1}{2}$. Let ω_c denote the counter rotation and recall $\frac{d\phi}{dt} = \omega_c + \omega_0$, so by simple algebra we wish to have $\frac{\omega_c}{\omega_0} = \frac{1}{\alpha+1}$. Additionally, recall

$$\omega_c = \frac{2I_{z'z'}}{I_{yy}} \omega_0 \sin\left(\frac{\theta}{2}\right)$$

And so the problem reduces to solving for θ in the following equation:

$$1 = (\alpha + 1)\frac{2I_{z'z'}}{I_{yy}} \sin\left(\frac{\theta}{2}\right)$$

However, I_{yy} is dependent on θ, so we must expand I_{yy} out into the form $I_{yy} = I_{y'y'} \cos^2(\theta/2) + I_{z'z'} \sin^2(\theta/2)$. Then by plugging this into the equation, we can convert everything into sines, apply the quadratic formula on $\sin(\theta/2)$ allowing us to solve for θ as:

$$\theta = 2 \arcsin\left(\frac{I_{z'z'}(\alpha + 1) + \sqrt{I_{z'z'}^2(\alpha + 1)^2 - I_{y'y'}(I_{z'z'} - I_{y'y'})}}{(I_{z'z'} - I_{y'y'})}\right)$$

where the substitution $\alpha = \frac{1}{2}$ gives the one-turn θ.