EULER'S ROTATION THEOREM: ROTATING OBJECTS IN 3-SPACE

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTERS OF SCIENCE

IN THE GRADUATE SCHOOL OF THE

TEXAS WOMAN'S UNIVERSITY

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCES

COLLEGE OF ARTS AND SCIENCES

BY

KASIE TAYLOR B.S.

DENTON, TEXAS

AUGUST 2014

TEXAS WOMAN'S UNIVERSITY
DENTON, TEXAS

July 10, 2014

To the Dean of the Graduate School:

I am submitting herewith a thesis written by Kasie Taylor entitled "Euler's Rotation Theorem: Rotating Objects in 3-Space." I have examined this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Masters of Science with a major in Mathematics.

Dr. Junalyn Navarra-Madsen, Major Professor

We have read this thesis and recommend its acceptance:

Dr. David Marshall

Dr. Don Edwards

Department Chair

Accepted:

Dean of the Graduate School

ACKNOWLEDGEMENTS

I would like to take this moment and acknowledge the many people who have contributed to this thesis. I would like to thank Dr. Karl Frinkle who first introduced me to this topic, and for challenging me to look at mathematics for the purpose of practical application instead of mindless equations and formulas that hold no purpose. I would like to thank my committee chair Dr. Junalyn Navarra-Madsen for her guidance; I would not have been able to complete my thesis without her constant help and direction. To my wonderful committee Dr. Don Edwards and Dr. David Marshall, your constructive comments and valuable feedback were essential to the preparation and finalization of this thesis. I am grateful to the faculty and staff of Texas Woman's University and Southeastern Oklahoma State University who through my bachelors degree and graduate degree they all challenged me to think outside my own box and explore areas that I never dreamed possible. I would also like to thank the Graduate Office of Texas Woman's University for their constant help in navigating paperwork, deadlines, and the numerous questions on formatting and procedure requirements for this thesis and graduation. Lastly, I would like to thank my family, for without your constant support, late night phone calls while driving home from class, and the encouragement to continue I would not have completed this thesis.

ABSTRACT

KASIE TAYLOR

EULER'S ROTATION THEOREM: ROTATING OBJECTS IN 3-SPACE

AUGUST 2014

The purpose of this thesis is to explore Euler's Rotation Theorem as it applies to the rotation of objects along various paths. Matrices can be used to represent these rotations along with the equation for a specific sphere. After these matrices are selected Maple programming will be used to calculate and further animate the rotation of a sphere (the earth) along an elliptical path, while another sphere (the moon) is rotating in a circular path around the first sphere. The computations in this paper were performed by using Maple™. Maple is a trademark of Waterloo Maple Inc. These rotations and the matrices that are yielded are known as orthogonal matrices. Even more specifically they can be thought of as special orthogonal matrices. This thesis investigates the various properties of these rotational matrices along with the relationship between orthogonal matrices and special orthogonal matrices.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

EULER'S ROTATION THEOREM

Introduction

Leonhard Euler was a Swiss mathematician, who was born 1707 in Basel and died 1783. Euler received his early formal education in Basel, and then at the age of thirteen he enrolled at the University of Basel. In 1723 he received his Master of Philosophy with a dissertation that compared the philosophies of Descartes and Newton. Euler received lessons from Johann Bernoulli who believed that Euler had an incredible talent for mathematics. Euler studied mathematics at the university level and later succeeded Daniel Bernoulli as the head of the mathematics department at The Academy at St. Petersburg (James, 2003, p.2).

Euler had many remarkable talents. After he lost his sight in his right eye in 1735 and in his left in 1766, he continued to publish the results of his studies by dictating them, largely aided by his phenomenal memory. Even though Euler was a very educated man he had the ability to communicate scientific matter effectively to a lay audience, a rare ability for a dedicated research scientist. This ability is still largely appreciated today when studying his many theorems (Beckmann, 1971, p. 143; Boyer, 1968, p. 482).

This thesis focuses on Euler's Rotation Theorem. In the realm of linear algebra, this theorem is known for its application to rotations of objects using rotation matrices and matrix multiplication. Jacques Philippe Marie Binet is credited with discovering the rules that allow matrices to be multiplied (Ikenaga, 2009). By combining Euler's Rotation Theorem and Binet's rules of matrix multiplication the amount of applications are endless. Chapter 1 focuses on the specifics of how to combine both of these ideas and then shows a specific application to better focus our understanding of Euler's Rotation Formula.

Rotation Matrix

A rotational matrix is defined as a matrix that is used to perform a rotation in Euclidean space (Shiskowski, 2001). These matrices can fall largely into two categories, those present in the 2-dimensional space real numbers (R^2) or those in the 3-dimensional space real numbers (R^3). That is not to say that rotational matrices do not exist outside of 2-dimensional and 3-dimensional spaces, because they do; yet the applications are not as easily visualized.

Since a rotational matrix is defined as being used to perform a rotation, then it must be said that there is more than one type of rotational matrix. The type of rotational matrix can be defined based on the pathway that the object follows. By adjusting the matrix slightly, a circular pathway matrix can easily be changed to reflect a rotation for an elliptical pathway. Both matrices are rotational matrices but each along a different pathway. Subsections 2-dimensional space and 3-dimensional space introduce both the circular and elliptical rotational matrices that occur in R^2 and R^3.

2-Dimensional Space

Circular

In R^2, the circular rotational matrix rotates a point some angle θ about the origin in the x-y Cartesian coordinate system. To derive the rotation matrix it is necessary to first look at the identity matrix $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Each column in I represents a vector that is used to describe a point by its x and y coordinate. Let $\vec{x} = \langle 1,0 \rangle$ and $\vec{y} = \langle 0,1 \rangle$ on the x-y Cartesian plane. Using the unit circle and prior knowledge of the trigonometric ratios such as sine, cosine, and tangent, the x coordinate can be described as $x = \langle \cos\theta, \sin\theta \rangle$ in which θ represents the angle of rotation about the origin. Likewise the y coordinate can be described as $y = \langle -\sin\theta, \cos\theta \rangle$ in which θ represents the angle of rotation about the origin. Thus the circular rotational matrix for R^2 is $R_C = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. To rotate a specific point using matrix multiplication in R^2 each point, Q, will need to be represented as a column vector as such, $Q = \begin{bmatrix} x \\ y \end{bmatrix}$. Then the point can be rotated, in a circular fashion, by performing the multiplication RQ=Q', where Q' is the image of Q after a rotation of angle θ about the origin (Shiskowski, 2011).

Elliptical

The elliptical rotation matrix is formulated using the same basic steps as for the circular pathway. Yet a different approach is needed. Suppose there is a point

$P=(x_1,y_1)$ such that $P=(a*\cos(\theta),b*\sin(\theta))$, where a is the semimajor axis and b is the semiminor axis and θ represents the angle at which P is, with respect to the positive x-axis. Now we rotate P, an angle of ϕ, such that, $P'=(a*\cos(\theta+\phi),b*\sin(\theta+\phi))$. By the trigonometric addition formula

$P'=(a*\cos(\phi)\cos(\theta)-a\sin(\phi)\sin(\theta),b*\sin(\phi)\cos(\theta)+b*\cos(\phi)\sin(\theta))$. By

substitution, $P'=\left(x_1\cos(\theta)-\frac{a}{b}y_1\sin(\theta),y_1\cos(\theta)+\frac{a}{b}x_1\sin(\theta)\right)$, yielding the elliptical

rotation matrix $R_E=\begin{bmatrix}\cos(\theta) & -\frac{a}{b}\sin(\theta)\\ \frac{b}{a}\sin(\theta) & \cos(\theta)\end{bmatrix}$. Now choose a point on the ellipse Q,

represented by $Q=\begin{bmatrix}x\\ y\end{bmatrix}$. Then the point can be rotated, in an elliptical fashion, by performing the multiplication $R_EQ=Q'$, where Q' is the image of Q after a rotation of angle θ about the origin (Shiskowski, 2011).

3-Dimensional Space

Circular

Similarly there exists a circular rotation matrix in R^3. There are some slight differences that are very important to point out. Since we are now in R^3, rotation will take place about an axis and just about a point as it did in R^2. Thus instead of 1 rotation matrix as there was in R^2, there are now 3 rotation matrices that exist in R^3. Each rotational matrix in R^3 can be found in much of the same fashion. So let us first look at the rotational matrix in which a point can be rotated about the x-axis. We must first look at the identity matrix for R^3 where $I^3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$. Let each column represent a vector that can be described by its x, y, and z coordinate. Let $\bar{x} = \langle 1,0,0 \rangle$, $\bar{y} = \langle 0,1,0 \rangle$, and $\bar{z} = \langle 0,0,1 \rangle$. Similar to R^2, we will use prior knowledge of trigonometric ratios and the unit circle to describe these vectors as a rotation about the x-axis. Since the rotation is not about the x-axis the x-coordinate will remain as is. Now following the process from R^2, while keeping in mind that the x coordinate will remain constant when rotating about the x-axis, $y = \langle 0, \cos\theta, \sin\theta \rangle$ and $z = \langle 0, -\sin\theta, \cos\theta \rangle$. Thus the rotation matrix for R^3 rotation about the x-axis is $R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$. Similarly the rotation matrices for rotation about the y-axis and z-axis, in which the y coordinate and z coordinate are constant respectively, are as follows, $R_y = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$ and

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Elliptical

Just as in R^2 where there was both circular and elliptical, the same holds true for R^3. The main differences between R^2 and R^3 that were explain for a circular rotation hold true for the elliptical pathway. Again the main difference is the choice of a for the semimajor axis and the choice of b for the semiminor axis. Thus following the same process as above and rotating with respect to an axis the three elliptical rotational matrices are as follows: $R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\frac{a}{b}\sin\theta \\ 0 & \frac{b}{a}\sin\theta & \cos\theta \end{bmatrix}$, $R_y = \begin{bmatrix} \cos\theta & 0 & -\frac{a}{b}\sin\theta \\ 0 & 1 & 0 \\ \frac{b}{a}\sin\theta & 0 & \cos\theta \end{bmatrix}$,

and $R_z = \begin{bmatrix} \cos\theta & -\frac{a}{b}\sin\theta & 0 \\ \frac{b}{a}\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

Euler's Rotation Theorem

Euler's Rotation Theorem states that in three dimensional space, any displacement of a rigid body such that a point on the rigid body remains fixed is equivalent to a single rotation about some axis that runs through the fixed point (Palais, 2007). This leads to the thought that any object can be rotated through any specific axis. The object would need to be a rigid body with a specific point that is fixed on that rigid body. A sphere is an example of such a rigid body. A sphere has a specific point, namely the center, which is fixed and can be tracked through each rotation. It is first important to understand that the best way to represent these rotations will be with vectors which will then be represented as a matrix.

First consider the rotation of a point Q, about some line L in which Q is not on L. As Q rotates about L, Q will travel along the arc of circle C orthogonal to L, such that the center of C is the closest point on L to Q. Vector operations and position vectors can be useful when determining C. C will be defined as perpendicular to line L, whose radius will be the shortest distance from Q to L. The circle C will be given parametrically in terms of an angle of rotation α. After a rotation of α, the new point will be defined as N. First find the center of C. This vector is found by computing the projection of vector Q onto the vector P. Vector P is defined by the point P being on line L and passes through the origin. To compute this projection uses the formula $C = \frac{P \cdot Q}{|P|^2} P$. The radius, R, of the circle is found by $R = |Q - C|$. Now consider 2 new vectors U and V. U is a vector that is of unit length, starting at C and traveling in the direction of Q. V is a scalar vector orthogonal to U and C, which is then made unit length by dividing by its own magnitude. U is defined as $U = \frac{Q - C}{R}$, and V is defined as $V = \frac{C \times U}{|C \times U|}$. Since both U and V are in the plane of C they can be thought of as the positive x-axis and y-axis respectively, where Q is positioned at (1,0) in the (U, V)-coordinate system. The circle of rotation now has a vector parametric equation of $Q_{rot(\alpha)} = C + R\cos(\alpha)U + R\sin(\alpha)V$ (Shiskowski, 2011).

Now that a point is easily rotated in circular and elliptical pathways, the same thought applies when rotating a sphere about a line. Since the sphere has a fixed center the same process applies. For specific applications, a particular path can be predetermined if path specifications are met.

Let us turn our attention to a more formal proof of Euler's fixed point theorem: the axis of a rotation. This specific theorem is particularly relevant to our rotations in IR^3. "Leonhard Euler stated in 1775, that in three dimensions, every rotation has an axis. Euler's original formulation of the result is that if a sphere is rigidly rotated about its center then there is a diameter that remains fixed. A modern reformation is: **Euler's Theorem**: *If* $\mathbf{R}$ *is a* 3 x 3 *matrix satisfying* $\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}$ *and* $\det\mathbf{R} = +1$, *then there is a non-zero vector* $\mathbf{v}$ *satisfying* $R\mathbf{v}=\mathbf{v}$ (Palais, 2007)." I will leave the proof to Chapter 3 when we take a more in-depth look at these matrices.

CHAPTER II

APPLICATION

Now that the basics of the Euler's rotation formula have been introduced I will now turn the topic to a specific example of how rotation matrices and matrix multiplication with the assistance of Maple Programming can be used to make a 3 dimensional model of the sun with the earth and moon rotating in two different pathways, one circular and the other elliptical. The computations in this paper were performed by using Maple™. Maple is a trademark of Waterloo Maple Inc.

Several subtle steps are necessary for this rotation to work. I will break each step down individually showing the matrices and the programming that make this work. I will first begin in R^2 focusing on the necessary rotations there so that I can build upon them to then move to R^3. But first, I will demonstrate a circular rotation of a point around the origin, a point about a point. Then I will demonstrate the elliptical rotation with a point about the origin, a point about a point. Once the basis of these rotations in R^2 is fully explained I will expand to R^3. Starting with the easier of the 2 types of rotations—the circular, I will explain how to rotate a sphere about the x-axis and then a sphere about a line not an axis. The last of the simple rotations I will demonstrate will include the elliptical rotation of a sphere about an axis with specified values for the major and minor axis. Finally I will combine both three dimensional rotations of sphere in a circular and elliptical paths and combine them in one final rotation to represent the rotation of the earth and moon about the sun. It is imperative that readers understand that this specific demonstration can be adjusted to represent many other rotations; this is just one of many examples that can be demonstrated.

Rotation of a Point

Circular About the Origin

First I will rotate a point about the origin. I choose the point Q to be (3,4) and point P to be the origin (0,0). I then choose based on these points that the center of my circle will be point P and the radius will be 5 such that Q will be a point on the circle. For programming purposes, it is easiest to parameterize the circle to display it on my grid and input the rotation matrix calling it "A". The rotation matrix A is the rotation matrix that was described earlier as the circular rotation matrix for I^2 $R_C = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$.

For the purposes of this specific example I have called this matrix $A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$, so as not to cause confusion. So $x = 5\sin(t)$ and $y = 5\cos(t)$ with t values ranging from -6 to 6.

Here is a visual of the path the point will take during its rotation.

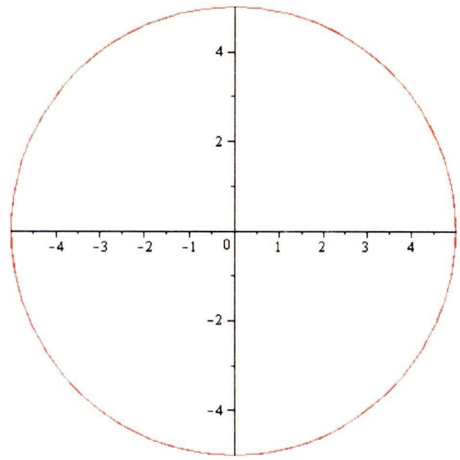


Figure 1: Circular pathway centered at the origin

Now to rotate the point multiply matrix "A" by point Q at various values of theta (0-2π) for a complete rotation. For Maple to perform this multiplication the point Q has been input as a vector matrix:

```
Q:=matrix(2,1,[3,4]);
```

$$Q := \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

This multiplication will yield a series of vectors that when animated will simulate the rotation along the circular path shown. The Maple format to animate the rotation is thus:

```
> Animation:=
animate(pointplot,[[T(theta)[1,1],T(theta)[2,1]]],theta=Pi/
8..16*Pi/8,frames=8,color=blue,symbol=solidcircle,symbolsiz
e=40):
```

With this, I can visually check to make sure my point Q is rotating along the circular path for theta values spanning the entire circle.

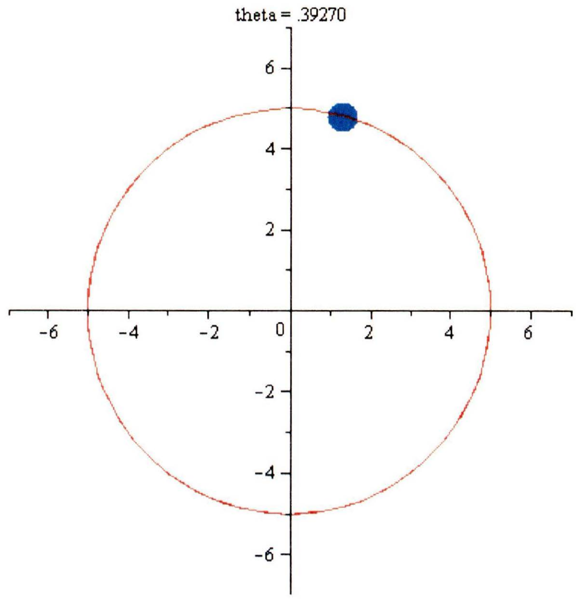


Figure 2: Point rotating about the origin in a circular motion

For the complete documentation of the coding that is required for this specific process please refer to Appendix A.

Circular About a Point Not the Origin

The next step is to rotate a point about a point not the origin in a circular motion. Once this point is chosen it is now the center of my circle of rotation. I chose that the center of my circle will be at the point P=(8,8). Then I choose a point on my circle to be Q=(13,8). With these points the equation of my circle is $(x-8)^2+(y-8)^2=25$. With the parameterization of the equation for the purpose of programming, we now have the equations $x=5\sin(t)+8$ and $y=5\cos(t)+8$, to represent our circle. The values of t to represent this circle are $t=-6$ to $t=6$.

Below is a visual representation of the path the point will take during this rotation.

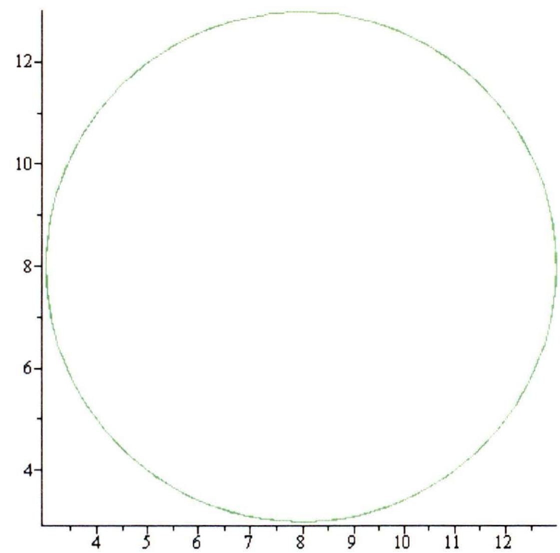


Figure 3: Circular pathway centered at (8,8)

To show the rotation of the point around this circular path, I will apply matrix multiplication between matrix "A" and the point Q at values of theta from 0 through 2π for a complete rotation. The rotation matrix A is the rotation matrix that was described earlier as the circular rotation matrix for R^2 $R_C = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. For the purposes of this specific example I have called this matrix $A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$, so as not to cause confusion. This is also the same matrix that was utilized in the rotation of the point about the origin in a circular motion.

For Maple to perform this multiplication the point Q has been input as a vector matrix:

```
Q:=matrix(2,1,[13,8]);
```

$$Q := \begin{bmatrix} 13 \\ 8 \end{bmatrix}$$

This matrix multiplication will yield a series of vectors that when animated will simulate the rotation along the circular path shown. Since the rotation matrix is best applied about the origin, there is a very specific bit of coding that must be included to get the appropriate vectors.

The basic idea is that since the rotation is occurring at a point which is not the origin there must be a shift built into each calculation that will shift the pathway, and point to the origin, rotate the point a set theta then shift the pathway and point back to the point we originally wished to rotate about. This shift is characterized in the Maple coding thus:

```
> T:= theta->evalm(A(theta)&*(Q-P)+P):
```

Basically what the code is saying is that for all values of theta from 0 to 2π, the matrix A should be multiplied by Q, but only after P is subtracted from Q $[13,8]-[8,8]=[5,0]$. This subtraction yields a vector whose center of rotation is at the origin since the radius of the circle of rotation was chosen to be 5. Now that the new vector is now centered at the origin, the multiplication with the rotation matrix occurs yielding a vector of a rotation of theta. Now the new vector [x,y] is then added with the vector P to shift the new vector so that the end result will be as if the vector was rotated about a point not centered at the origin. This shift occurs for each theta from 0 to 2π. Once each of vectors is calculated it is now necessary to animate the rotation with help of Maple. The Maple format to animate the rotation is:

```
> Animation:=
animate(pointplot,[[T(theta)[1,1],T(theta)[2,1]]],theta=0..
2*Pi,frames=30,color=blue,symbol=solidcircle,symbolsize=40)
:
```

With these vectors in animation, I can visually check to make sure my point Q is rotating along the circular path for theta values spanning the entire circle.

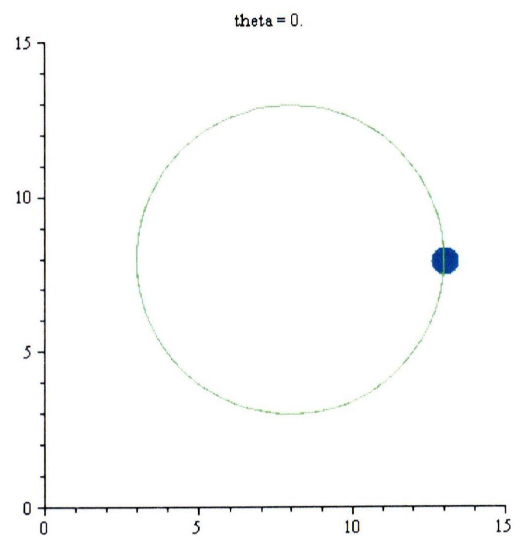


Figure 4: Point Rotating about (8,8) in a circular motion

For the complete documentation of the coding that is required for this specific process please refer to Appendix B.

Elliptical Rotation About the Origin

Now the rotation of a point is complete in the circular path both at the origin and a point not the origin, I will now continue in R^2 with the rotations of a point in the elliptical motion first about the origin then I will rotate about a point not the origin. But first I will rotate a point in an elliptical motion about the origin. So I first choose an ellipse centered at the origin with specified major axis and minor axis measurements of my choosing. So I will choose an ellipse that is centered about the origin with a major axis measurement of 5 and a minor axis measurement of 3. Again for programming purposes it is easiest to represent this ellipse in parameterization form. Thus the ellipse is represented by the following equations: $x = 5\cos(t)$ and $y = 3\sin(t)$. A visualization of the path of rotation can be seen below with values of t ranging from $t = -6$ to $t = 6$.

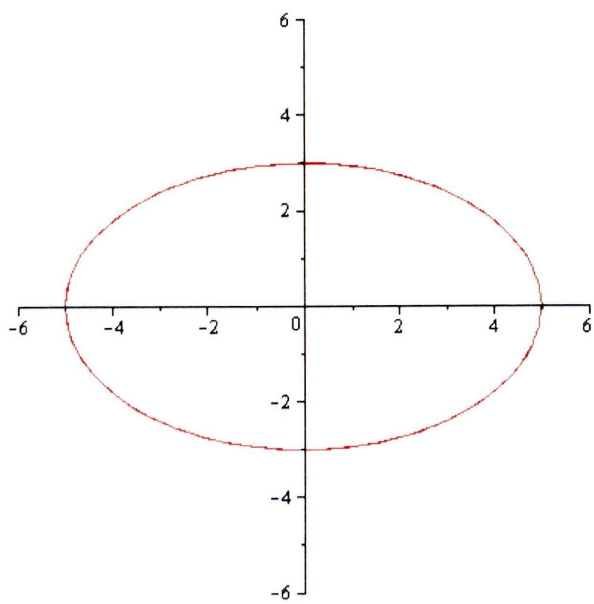


Figure 5:. Elliptical pathway centered at the origin

To perform the rotation necessary, a new rotation matrix must be utilized. This rotation matrix is specific to the rotation in an elliptical pattern in R^2. It is the same rotation matrix that was demonstrated earlier; $R_E = \begin{bmatrix} \cos(\theta) & -\frac{a}{b}\sin(\theta) \\ \frac{b}{a}\sin(\theta) & \cos(\theta) \end{bmatrix}$. For the purposes of this specific elliptical rotation I have chosen to represent this matrix as,

$A = \begin{bmatrix} \cos(\theta) & -\frac{a}{b}\sin(\theta) \\ \frac{b}{a}\sin(\theta) & \cos(\theta) \end{bmatrix}$. This matrix has values of a and b these values correspond to the values of the major and minor axis of the path of rotation. So before rotating these values must be specified as part of the programming to insure that the rotation is executed correctly.

I now choose a point Q on the ellipse $Q = (5,0)$. Then for the purposes of programming I input this point as a matrix and represent it as a vector, as follows:

```
> Q:=matrix(2,1,[5,0]);
```

$$Q := \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

Now to rotation point Q about the ellipse, $x = 5\cos(t)$ $y = 3\sin(t)$, I perform matrix multiplication between the vector Q and the rotation matrix A with values of theta ranging from 0 to 2π to show a complete rotation. The multiplication will yield a series of vectors that when animated will simulate the rotation of point Q about the ellipse. The specific Maple command to yield this animation is as follows:

```
> Animation:= animate(pointplot, [[T(theta)[1,1],
T(theta)[2,1]]], theta = 0..2*Pi, frames = 20, color =
blue, symbol=solidcircle, symbolsize=40):
```

With this animation it is quite easy to see if point is truly rotating about the ellipse in a complete rotation. It is imperative to check these small rotations each time. By checking each smaller rotation to make sure the coding and the multiplication are yielding correct vectors, we can then expand our reach each time to a new dimension or a new rotation point.

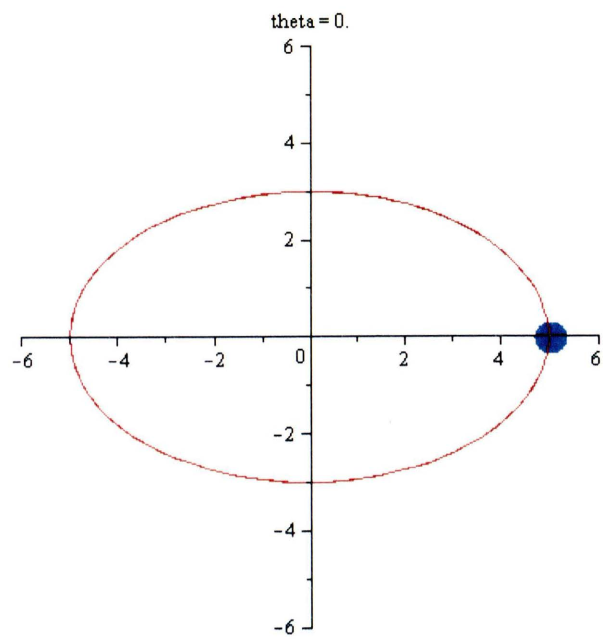


Figure 6: Point rotating about the origin in an elliptical motion

For a complete listing of the necessary coding for this specific rotation please refer to Appendix C.

Elliptical Rotation About a Point Not the Origin

Now to finish up the R^2 rotations so that we can build a firm foundation before rotating in R^3, I will now rotate a point about a point not centered at the origin in an elliptical motion.

For this rotation I will start by choosing a center of rotation and a path on which the rotation will occur. The center of rotation I will choose to be $P = (4,4)$ and the equation for my pathway of rotation to be $\frac{x^2}{25} + \frac{y^2}{9} = 1$. Again it is very important to the pathway of rotation. Thus the pathway can be represented by, $x = 5\cos(t) + 4$ and $y = 3\sin(t) + 4$. To visualize the rotational pathway I have chosen to allow Maple to plot the ellipse with values of t ranging from $t = -6$ to $t = 6$.

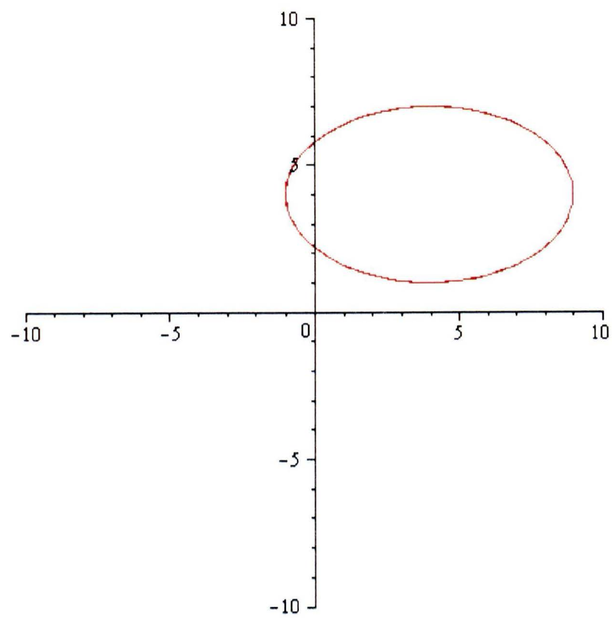


Figure 7: Elliptical pathway centered at (4,4)

To perform the rotation necessary, a new rotation matrix much be utilized. This rotation matrix is specific to the rotation in an elliptical pattern in R^2. It is the same rotation matrix that was demonstrated earlier; $R_E = \begin{bmatrix} \cos(\theta) & -\frac{a}{b}\sin(\theta) \\ \frac{b}{a}\sin(\theta) & \cos(\theta) \end{bmatrix}$. For the purposes of this specific elliptical rotation I have chosen to represent this matrix as,

$A = \begin{bmatrix} \cos(\theta) & -\frac{a}{b}\sin(\theta) \\ \frac{b}{a}\sin(\theta) & \cos(\theta) \end{bmatrix}$. I now choose a point Q on the ellipse $Q = (9,4)$. Then for the purposes of programming; I input this point as a matrix and represent it as a vector, as follows:

```
> Q:=matrix(2,1,[9,4]);
```

$$Q := \begin{bmatrix} 9 \\ 4 \end{bmatrix}$$

Now to rotate point Q about the ellipse, $x = 5\cos(t) + 4 \quad y = 3\sin(t) + 4$, I perform matrix multiplication between the vector Q and the rotation matrix A with values of theta ranging from 0 to 2π to show a complete rotation. Since the rotation matrix is best applied about the origin; there is a very specific bit of coding that must be included to get the appropriate vectors.

The basic idea is that since the rotation is occurring at a point not the origin there must be a shift built into each calculation that will shift the pathway and point to the origin, rotate the point a set theta then shift the pathway and point back to the point we originally wished to rotate about. This shift is characterized in the Maple coding thus:

```
> T:= theta->evalm(A(theta)&*(Q-P)+P):
```

Basically what the code is saying that for all values of theta from 0 to 2π, the matrix A should be multiplied by Q, but only after P is subtracted from Q; $[9,4]-[4,4]=[5,0]$. This subtraction yields a vector whose center of rotation is at the origin since the major axis of the ellipse of rotation was chosen to be 5. Now that the new vector is now centered at the origin, the multiplication with the rotation matrix occurs yielding a vector of a rotation of theta. Now the new vector [x,y] is then added with the vector P to shift the new vector so that the end result will be as if the vector was rotated about a point not centered at the origin. This shift occurs for each theta from 0 to 2π. Once each of vectors is calculated it is now necessary to animate the rotation with help of Maple. The Maple format to animate the rotation is thus:

```
> Animation:= animate(pointplot, [[T(theta)[1,1],
T(theta)[2,1]]], theta = 0..2*Pi, frames = 20, color =
blue, symbol=solidcircle, symbolsize=40):
```

With these vectors in animation; I am able to check to make sure that the point is rotating properly along the pathway specified.

Below is a visualization of what the rotation will look like at a specified theta.

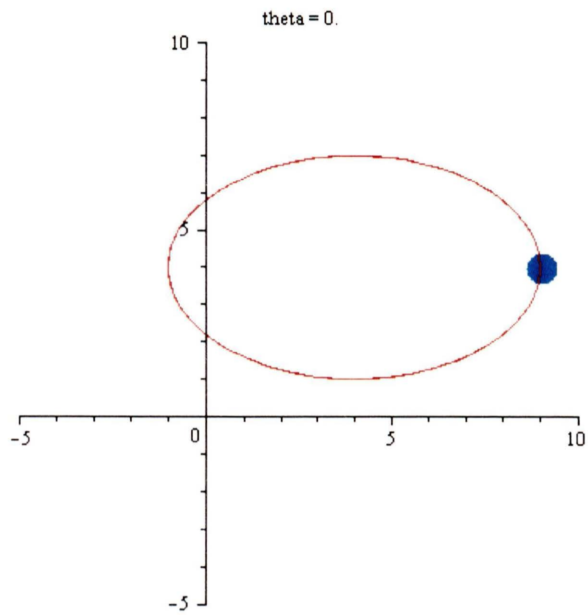


Figure 8: Point rotating about an ellipse centered at (4,4)

For a complete listing of the necessary coding for this specific rotation please refer to Appendix D.

Rotation of a Sphere

Circular About an Axis

Now that the rotations in R^2 are completed it is time to turn our attention to the rotations in R^3. As described previously in the discussion of Euler's Rotation Formula, it is easiest to rotate in R^3 with vector projection. So let me do a quick review of how vector projection will work, and I will apply it to this specific example.

To start, I will first choose a sphere given by the parameterization of $x = 1\sin(\varphi)\sin(\theta) + 6$, $y = 1\sin(\varphi)\cos(\theta) + 3$, and $z = 1\cos(\varphi) - 4$. For programming purposes, I will represent this sphere as a single vector t, where

$t = [1\sin(\phi)\sin(\theta) + 6, 1\sin(\phi)\cos(\theta) + 3, 1\cos(\phi) - 4]$. This sphere is seen as

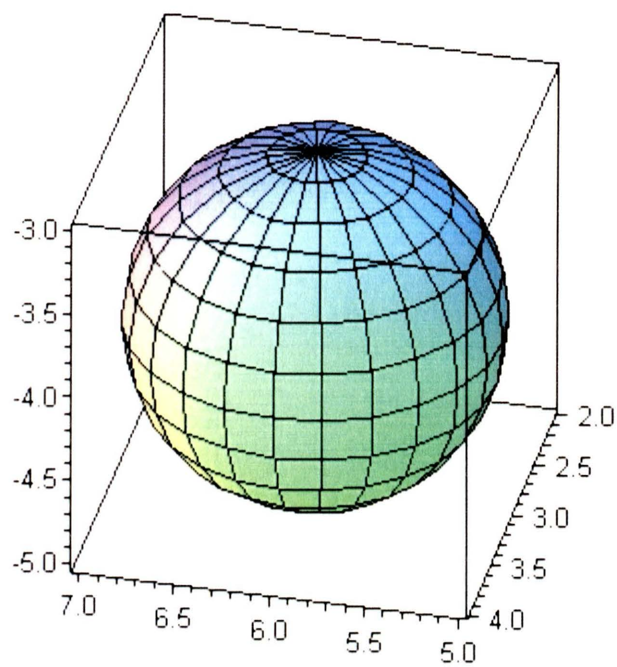


Figure 9: Sphere to be rotated

I will then choose an axis of rotation which will be the x-axis. I call this vector P and represent it as $p = [1,0,0]$.

Now that I have chosen the sphere to rotate as well as the axis of rotation, I will find a vector C which is the center of the circle of rotation. Vector C is found by computing the projection of the vector Q (our sphere, which in the Maple coding is referred to as *t*) onto the vector P (parallel to our line through the origin). Since the x-axis is already a line through the origin I can use $p = [1,0,0]$ as my vector, without it being necessary to find a separate vector. Using the vector projection formula yields:

$C = \frac{P \cdot Q \cdot P}{|P|^2}$. So for our specific sphere and axis of rotation the center of rotation

occurs at $c = [6 + \overline{\sin(\phi)\sin(\theta)},0,0]$. You will notice that on this very simple calculation the center of rotation is still in terms of ϕ and θ. As the calculations proceed the vectors will continue to be in terms ϕ and θ until the final command where specific values of ϕ and θ are given so as to animate the rotation. Thus from here I will concentrate of the review of how to calculate the vectors that are necessary for the rotation (Shiskowski, 2011).

The radius R, of the circle of rotation, is just the length of the vector $Q - C$, and so $R = |Q - C|$. Next, consider the following vector: $U = \frac{Q - C}{R}$ which is of unit length, starting at C and travelling in the direction of Q. The vector U is in the plane of the circle of rotation. Another unit vector in the plane of the circle of rotation, perpendicular to both U and C, is given by $V = \frac{C \times U}{|C \times U|}$. V is simply a scalar multiple of $C \times U$, and must therefore be perpendicular to both C and U, by definition of the cross product. V is made unit length simply by dividing the cross product by its magnitude (Shiskowski, 2011).

So what has this accomplished? In the plane of the circle of rotation, U and V are acting as the positive x and y axes unit vectors where our point Q is always positioned at the coordinates (1, 0) in the (U,V) coordinate system. Now the circle of rotation has the vector parametric equation given by: $Q_{rot}(\alpha) = C + R\cos(\alpha) + R\sin(\alpha)V$.

Now to animate this rotation I input the calculated values of C, R, and V along with values of ϕ from 0 to π, values of θ from 0 to 2π, and values of A from 0 to 6. With these specific values it ensures a complete rotation about the x-axis. To check the rotation of the sphere about the x-axis the following Maple commands are input:

```
> sprot:=A-
>evalf(convert(evalm(c+r*cos(A)*U+r*sin(A)*V),list)):
> movingsp:=animate3d(sprot(A), phi = 0..Pi, theta =
0..2*Pi, A = 0..6,frames=20):
```

yielding the following diagram, where the y axis is vertical, the z-axis is coming out of the paper, and the x-axis is horizontal.

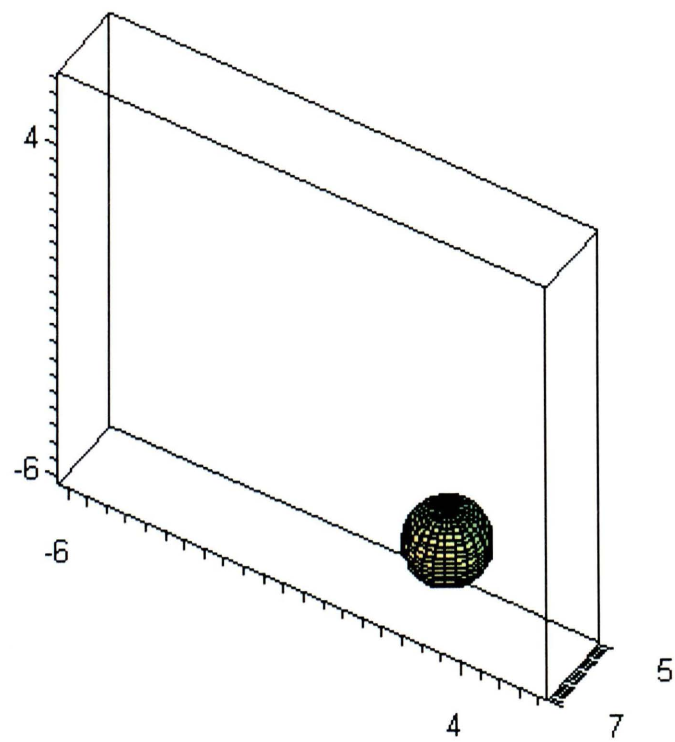


Figure 10: Sphere rotating along a circular path about an axis

For a complete listing of the necessary coding for this specific rotation please refer to Appendix E.

Circular About a Point Not on an Axis

Now that the rotation in R^3 has been demonstrated with a circular rotation about an axis it is time to turn our attention to rotating the object about a point that is not on a specific axis. When rotating in R^3 it is important to remember that we will be rotating about a line that exists through the point that we would like to rotate about.

Since we rotate in R^3 with the use of vector projection it is only necessary to change the point of rotation to a line containing the point we wish to rotate about. Then the process described above will allow us to rotate a sphere in I^3 about a line containing a specific point of my choosing.

So to begin rotating I will first choose the sphere I wish to rotate. Again for the purposes of programming it is imperative that the equation parameterized as such $x = 1\sin(\varphi)\sin(\theta) + 6$, $y = 1\sin(\varphi)\cos(\theta) + 3$, and $z = 1\cos(\varphi) - 4$. Once the sphere is parameterized I then represent it as a single vector *t,* where $t = [1\sin(\phi)\sin(\theta) + 6, 1\sin(\phi)\cos(\theta) + 3, 1\cos(\phi) - 4]$.

An example of this sphere is seen below:

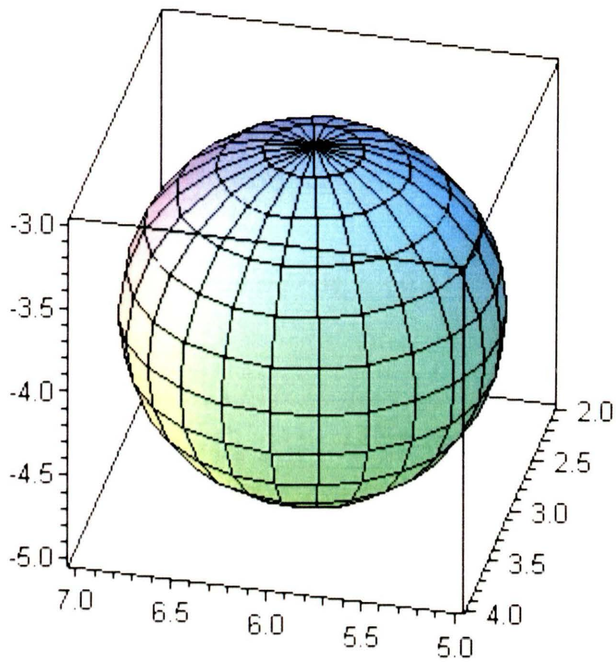


Figure 11: Sphere to be rotated

Now that I have chosen the sphere that I wish to rotate, I now choose the point that I wish to rotate about. I choose this point to be (3,4,5). Now I will represent this point as a vector to allow Maple to do the calculations correctly. This vector will be represented as $p = [3,4,3]$.

Since the vector projection process has recently been explained I will utilize key pieces to apply vector projection to this specific example. The first step once selecting the vector, *p,* and the sphere I wish to rotate, is compute the center of rotation using $C = \dfrac{P \cdot Q \cdot P}{|P|^2}$, where Q is representing the sphere (Shiskowski, 2011). Thus after Maple does the calculations the center of rotation is

$$c := \left[\frac{3}{5} + \frac{9}{50}\overline{\sin(\phi)\sin(\theta)} + \frac{6}{25}\overline{\sin(\phi)\cos(\theta)} + \frac{3}{10}\cos(\overline{\phi}), \frac{4}{5} + \frac{6}{25}\overline{\sin(\phi)\sin(\theta)} + \frac{8}{25}\overline{\sin(\phi)\cos(\theta)}\right.$$

$$\left. + \frac{2}{5}\cos(\overline{\phi}), 1 + \frac{3}{10}\overline{\sin(\phi)\sin(\theta)} + \frac{2}{5}\overline{\sin(\phi)\cos(\theta)} + \frac{1}{2}\cos(\overline{\phi})\right]$$

Notice again that the center of rotation is still in terms of ϕ and θ. This will be the case for each of the proceeding calculations of vectors R, U, and V. After each of these vectors are calculated I have specific calculations for C, R, and V, which I will use in the equation $Q_{rot}(\alpha) = C + R\cos(\alpha) + R\sin(\alpha)V$ to rotate my sphere (seen as Q is the previous equation). I will rotate the sphere using values of ϕ from 0 to π and values of θ from 0 to 2π to collect vectors that will simulate a rotation when animated. With these vectors and values of A from 0-6, to show a complete revolution I use the following command to animate the rotation of my sphere about the point (3,4,5):

```
> sprot:=A-
>evalf(convert(evalm(c+r*cos(A)*U+r*sin(A)*V),list)):
> movingsp:=animate3d(sprot(A), phi = 0..Pi, theta =
0..2*Pi, A = 0..6, frames=20):
```

After I input these commands I can check my rotation to make sure I make a complete rotation and I am rotating about the point I have chosen.

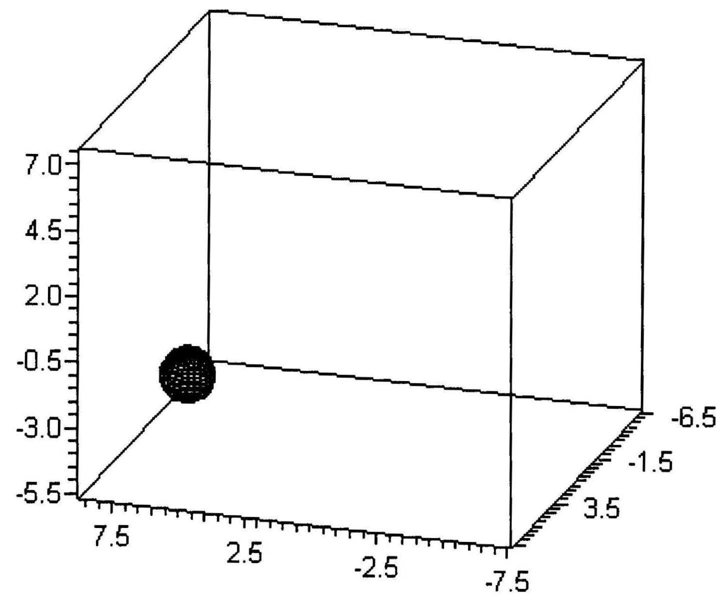


Figure 12: Sphere rotating along a circular path not on an axis

For a complete listing of the necessary coding for this specific rotation please refer to Appendix F.

Elliptical Rotation About an Axis

Before we continue on our journey with our specific application of rotation matrices, let us review what we have covered thus far. We started this journey with the intent of being able to utilize Maple Programming to make a 3 dimensional model of the sun with the earth and moon rotating in two different pathways, one circular and the other elliptical. Thus far I have rotated a point in a circular and an elliptical pathway, both at the origin and away from the origin in R^2, respectively. After these basic R^2 rotations I turned my attention to R^3, and then focused on the rotation of a sphere in a circular motion. The rotation of a circular pathway is very simple to calculate when using vector projection. This is possible since a circle is very symmetric and a rotation of $\frac{\pi}{4}$ for each coordinate will result in the coordinate remaining on the pathway. Now the last minor step before rotating with both pathways simultaneously is to rotate a sphere in R^3 along an elliptical pathway about an axis.

Since an ellipse does not have infinitely many lines of symmetry like that of a circle, I will focus on the idea that Euler suggested that it would be necessary to track the center of the sphere through each rotation of a specific degree. Tracking the center of the sphere is most logical given that with a rotation of $\frac{\pi}{4}$ would result in different coordinates for the x-coordinate and the y-coordinate, thus the rotation would not move to exactly $\frac{\pi}{4}$ on the ellipse. So when tracking the centers I will use the rotation matrix and matrix multiplication as previously done in R^2.

Before I can start multiplying I first choose a sphere and axis about which I would like to rotate. I choose a sphere with the center at (0,10,0) and having a radius of ½ . Thus the equation for my sphere is input as

$s := \frac{1}{2}\sin(\phi)\sin(\theta), 10 + \frac{1}{2}\sin(\phi)\cos(\theta), \frac{1}{2}\cos(\phi)$. Then values of ϕ ranging from $-\pi$ to π and values of θ ranging from $-\pi$ to π are input along with the coordinate axis of x, y, and z to get a visual of the sphere on the axis.

This visual can be seen as such:

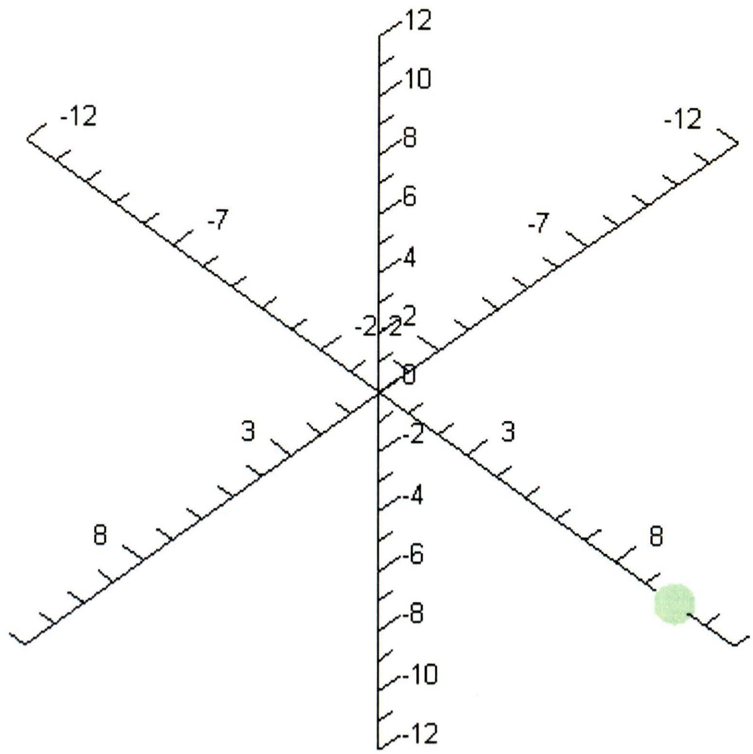


Figure 13: Sphere to be rotated

Now I will choose my pathway of rotation. I choose to have a rotation about the x-axis in an elliptical pathway with a major axis measurement of 10 and a minor axis measurement of 5. Please recall that the rotation matrix of an elliptical pathway for the x-axis is $R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\frac{a}{b}\sin\theta \\ 0 & \frac{b}{a}\sin\theta & \cos\theta \end{bmatrix}$. My chosen values for the major and minor axis correspond to the values of *a* and *b,* in the rotation matrix respectively. So for this specific example the rotation matrix that will be utilized is $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -2\sin t \\ 0 & \frac{1}{2}\sin t & \cos t \end{bmatrix}$.

Now that I have the appropriate rotation matrix for my specific example, I need to simply multiply my matrix by the center of my chosen sphere with values of t to span an entire rotation. Before continuing much further, I want to verify that this simple multiplication will yield a sphere whose center is along my pathway. I use the following Maple commands to check this multiplication:

```
> Q:= t -> evalm(A(t)&*Cr):
> Q(7*Pi/8);
```

$$\left[0 \quad -10\cos\left(\frac{1}{8}\pi\right) \quad 5\sin\left(\frac{1}{8}\pi\right) \right]$$

This shows that the rotation matrix A multiplied by the center of the sphere with the angle of $\frac{7\pi}{8}$ will result with a sphere whose center is at the point

$\left[0, -10\cos\left(\frac{1}{8}\pi\right), 5\sin\left(\frac{1}{8}\pi\right)\right]$.

Now that I am certain that this sphere is along my pathway I can calculate each center of the sphere with a sequence command with values of t from 0 to 2π for a complete rotation.

The Maple command used to calculate this sequence is given by

```
> SpherePlots:=
seq(plot3d([Q(k*Pi/6.)[1]+rad*sin(phi)*sin(theta),Q(k*Pi/6.
)[2]+rad*sin(phi)*cos(theta),Q(k*Pi/6.)[3]+rad*cos(phi)],
phi=-Pi..Pi, theta=-Pi..Pi), k=0..12):
```

Then after inputting the command to display the sequence, along with the elliptical pathway, and the x, y, and z axis ranging from -12 to 12 for each axis, the visual is as such:

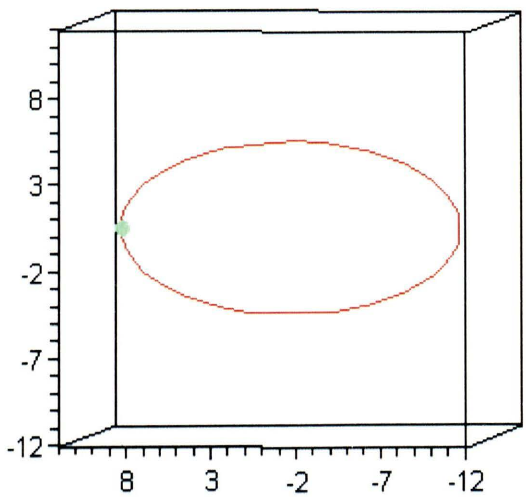


Figure 14: Sphere rotating about an ellipse

With this display in Maple, the animation of the sphere can be demonstrated for the rotation in I^3 along the elliptical pathway shown.

Now that the ground work has been laid to being able to model the sun with the earth and moon rotating in two different pathways, I will want to check my proportions to make sure that I can simulate a sun with the elliptical pathway that I have chosen. I plan to demonstrate a sun remaining stationary while the earth moves in an elliptical pathway and a moon traveling around the earth in a circular pathway. So before continuing on and merging the two pathways I will check proportions and set the sphere to represent the sun in my diagram.

So I will center the sphere to represent the sun at the foci (0,5,0) with a radius of 2. Below is the visual of the sun, the elliptical pathway, and the earth as it will rotate around the sun in the elliptical pathway shown.

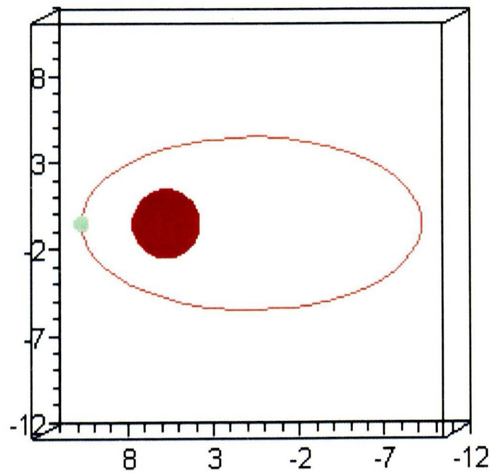


Figure 15: Position of the sun with the earth rotating about the elliptical path

For a complete listing of the necessary coding for this specific rotation please refer to Appendix G.

Rotation of a Sphere Along Two Pathways

Rotation of the Earth in an Elliptical Pathway While the Moon Rotates in a Circular Pathway Around the Earth Simultaneously

The final step is to be able to simulate the rotations of the earth and moon around the sun in an elliptical and circular pathway. I again will use Euler's idea of tracking the centers of the spheres as I rotate each sphere in these pathways simultaneously. To accomplish this it will be very important to be specific and precise within my programming to be able to track each sphere and its respective rotations. To begin I will choose the radius of each sphere and plot each of them on an axis to achieve a baring of their relationship to each other. I will parameterize each sphere for the purposes of programming. The earth will be centered at (0,10,0) with a radius of ½ and the moon will be centered at (0,11,0) with a radius of ¼ . Thus the parameterized equation for the sphere representing the earth is $x = 0 + \frac{1}{2}\sin(\phi)\sin(\theta)$, $y = 10 + \frac{1}{2}\sin(\phi)\cos(\theta)$, and $z = 0 + \frac{1}{2}\cos(\phi)$. The parameterized equations to represent the moon are $x = 0 + \frac{1}{4}\sin(\phi)\sin(\theta)$, $y = 10 + \frac{1}{4}\sin(\phi)\cos(\theta)$, and $z = 0 + \frac{1}{4}\cos(\phi)$.

Hence when entering the code into Maple it can be seen as:

```
> CrE:=[0,10,0]:
CrM:=[0,11,0]:
> radE:=1/2:
radM:=1/4:
>
earth:=[CrE[1]+radE*sin(phi)*sin(theta),CrE[2]+radE*sin(phi
)*cos(theta),CrE[3]+radE*cos(phi)]:
moon:=[CrM[1]+radM*sin(phi)*sin(theta),CrM[2]+radM*sin(phi)
*cos(theta),CrM[3]+radM*cos(phi)]:
```

Thus the display of these spheres is:

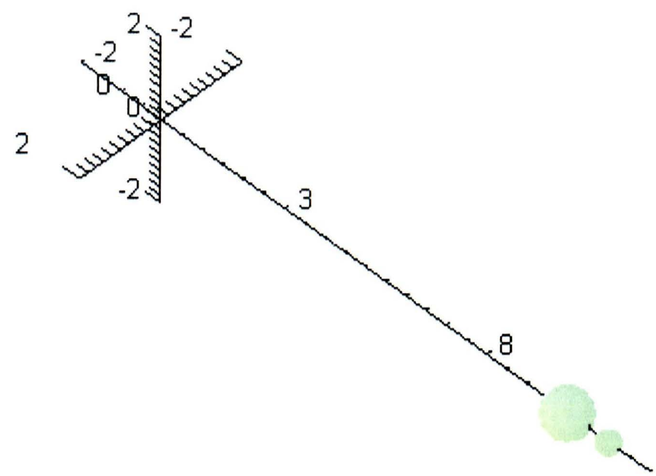


Figure 16: Earth sphere and Moon sphere on the x-axis

With this display a clear visual can be gained that is a representation between the sphere that represents the earth and the sphere that represents the moon. The next step is to create the specific rotational matrices for each the elliptical and circular rotations. This could be very delicate. I need to make sure that as the moon sphere rotates in a circular path that the radius of the rotation pathway is large enough that the moon sphere will not collide with the earth sphere during the rotation. To avoid this complication, I choose the position of the earth and moon spheres at (0,10,0) and (0,11,0) with radii of ½ and ¼ respectively. With these specifications on the spheres along with the chosen elliptical values of $a=10$ and $b=7$ along with a standard circular rotation about an axis the rotations should not collide. To check this theory first input the rotational matrix for the elliptical pathway as $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\frac{10}{7}\sin(t) \\ 0 & \frac{7}{10}\sin(t) & \cos(t) \end{bmatrix}$ and the circular pathway rotational matrix as $B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\sin(t) \\ 0 & \sin(t) & \cos(t) \end{bmatrix}$. Now I will input a specific value for t to test if the sphere will collide or not. This specific value is $t = \frac{7\pi}{8}$. At this value of t the position of the earth sphere's center is at $\left[0,-10\cos\left(\frac{1}{8}\pi\right),7\sin\left(\frac{1}{8}\pi\right)\right]$ and the position of the moon sphere's center is at $\left[0,-\cos\left(\frac{1}{8}\pi\right),\sin\left(\frac{1}{8}\pi\right)\right]$. After this verification I can conclude that with these specific rotational matrices and the specifics spheres I have chosen that they will rotate without collision.

Even after this verification there is yet another problem that presents itself. When rotating with two different pathways that are not symmetrical, or even similar in their rotation methods, when rotating at a value of $t = \frac{7\pi}{8}$, the spheres are not in the specific relationship that I wish to achieve. This was seen with the values that were presented above. The spheres did not collide but they were not in the same relationship to one another as when I began. Thus there is yet another step in the process that is required.

This step allows both spheres to rotate simultaneously and keep a consistent relationship with respect to each other. To achieve this first employ the idea that even though the elliptical and circular rotations are very different the final rotation that we wish to achieve does have some overlap within the two rotations. This overlap occurs because as the sphere representing the earth rotates about the sun in an elliptical fashion and the sphere representing the moon rotates around the earth in a circular path, I can conclude that the sphere representing the moon is also rotating about the sun in an elliptical path as well. Hence, both spheres representing the moon and earth are essentially rotating about the sun in the elliptical fashion. Since I also want the sphere representing the moon to rotate in a circular pathway about the sphere representing the earth, I will first rotate it in an elliptical fashion then do an additional rotation on the sphere representing the moon for the circular pathway. For example, to rotate both the sphere representing the earth and the sphere representing the moon by multiplying the centers by the elliptical rotation matrix $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\frac{10}{7}\sin(t) \\ 0 & \frac{7}{10}\sin(t) & \cos(t) \end{bmatrix}$ for a specific value of t. Then taking the resultant from the sphere representing the moon and multiply it by the circular rotation matrix $B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\sin(t) \\ 0 & \sin(t) & \cos(t) \end{bmatrix}$, to yield the desired final rotation.

Since I wish to show this as an animation, it is necessary to input many values of t and then loop them to continually to show this animation. Because this animation is more complicated than the other animations that have been attempted, it will be more useful to program this animation in a 'for loop'. Below is an example of one of a 'for loop' that was used in the Maple coding for this specific application.

```
> n:=100:
> for k from 1 to n do
 Earths[k+1]:=
[Q(2.*k*Pi/n)[1]+radE*sin(phi)*sin(theta),Q(2.*k*Pi/n)[2]+r
adE*sin(phi)*cos(theta),Q(2.*k*Pi/n)[3]+radE*cos(phi)]:
 Moons[k+1]:=
([Q(2.*k*Pi/n)[1],Q(2.*k*Pi/n)[2],Q(2.*k*Pi/n)[3]]+[R(18.*k
*Pi/n)[1]+radM*sin(phi)*sin(theta),R(18.*k*Pi/n)[2]+radM*si
n(phi)*cos(theta),R(18.*k*Pi/n)[3]+radM*cos(phi)]):

MoonOrbit[k+1]:=[Q(2.*k*Pi/n)[1],Q(2.*k*Pi/n)[2]+EMrad*cos(
t),Q(2.*k*Pi/n)[3]+EMrad*sin(t)]:
end do:
```

After the 'for loop' has been processed for each value of n, another 'for loop' is executed to display each value as a three dimensional plot. Then the last step to this process is to execute the command to display each plot with its spacecurve. The spacecuve plots allow the viewer to see each rotational pathway as the spheres rotate about the sphere represented by the sun. Below is the display of the final animation at different stages of the final animation.

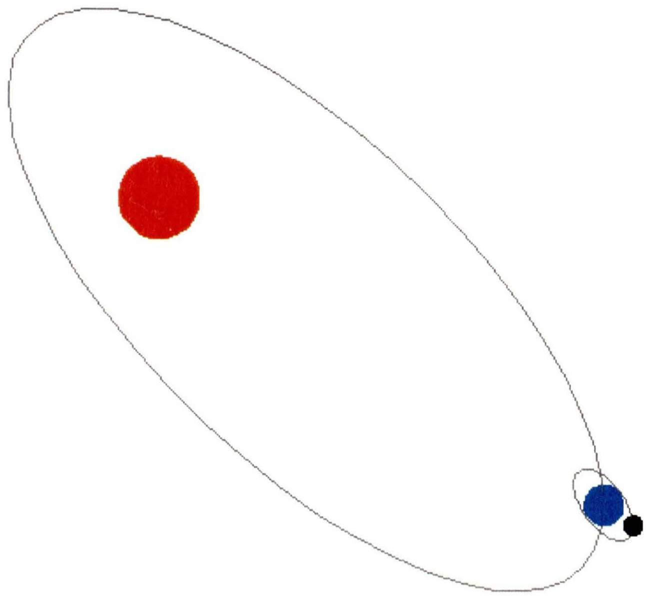


Figure 17: Rotating earth and moon about the sun-Stage 1 (n=1)

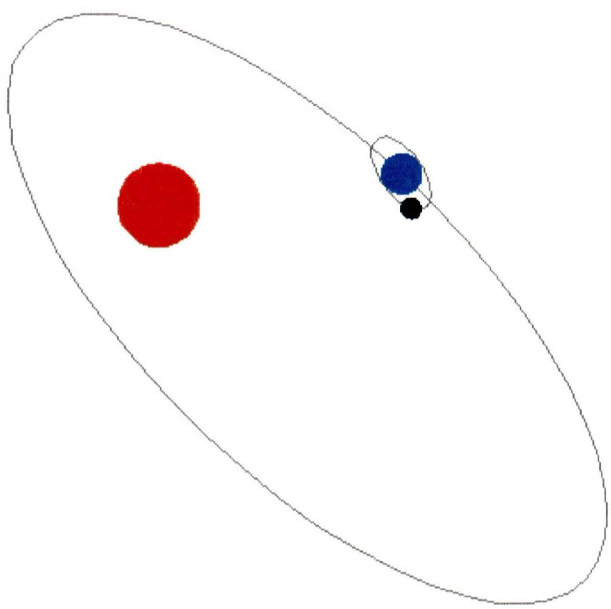


Figure 18: Rotating earth and moon about the sun-Stage 2 (n=21)

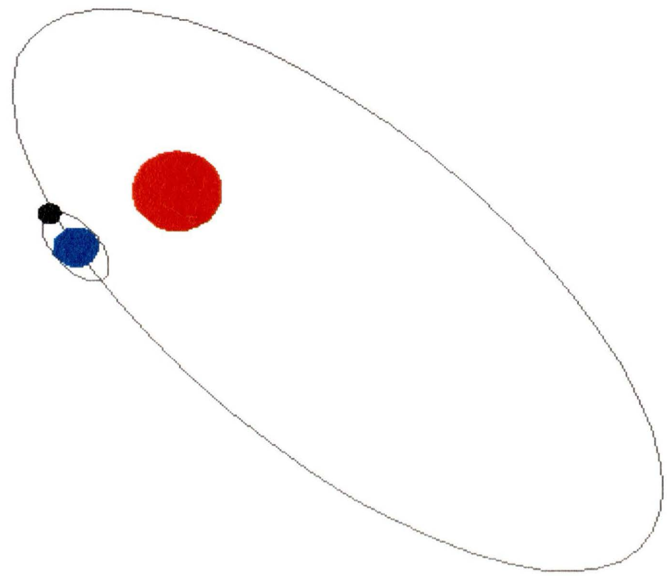


Figure 19: Rotating earth and moon about the sun-Stage 3 (n=61)

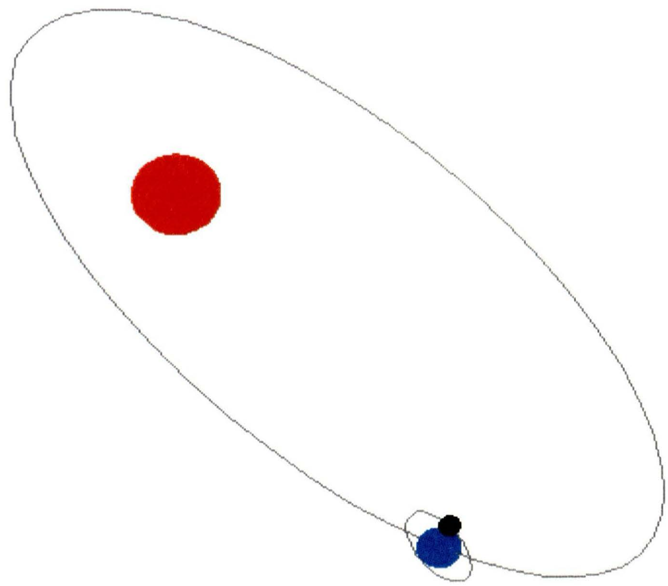


Figure 20: Rotating earth and moon about the sun-Stage 4 (n=81)

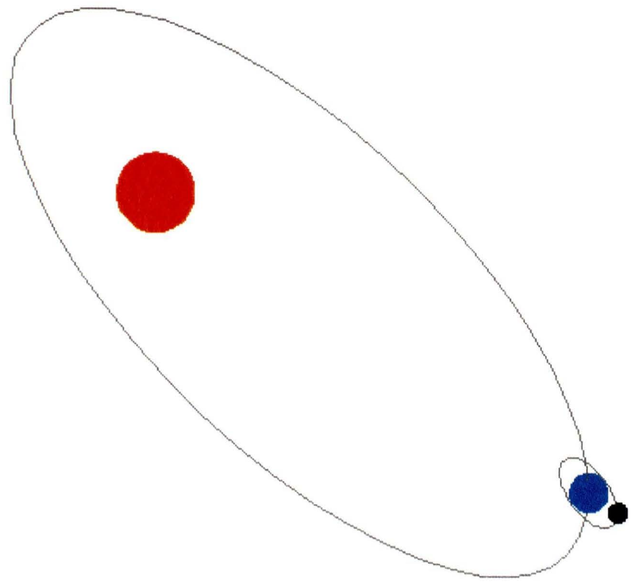


Figure 21: Rotating earth and moon about the sun-Stage 5 (n=101)

For a complete listing of the necessary coding for this specific rotation please refer to Appendix H.

CHAPTER III

ORTHOGONAL MATRICES

Now that the basis for Euler's Rotation Formula has been explored with an application of how this formula can be used in the world around us, I will turn my attention to the matrices themselves that are used as part of Euler's Rotation Formula in this application. Each of the rotational matrices used are a specific type of matrix called an orthogonal matrix. An orthogonal matrix is a square matrix in which its transpose is equal to its inverse. For example, given a matrix Q, Q is said to be orthogonal if $Q^T = Q^{-1}$. Hence $Q^T Q = QQ^T = I$ (Todd).

If I show that the rotation matrix used is an orthogonal matrix then all the properties that apply to a general orthogonal matrix will also apply to my specific rotational matrices. So first I will show that all orthogonal matrices make a group, and then show that the matrices used within my application are also orthogonal matrices, making these rotational matrices within the application are a part of the orthogonal matrix group.

Orthogonal Matrix Group

Now consider the set of all orthogonal matrices. To show that this set is a group we must first understand what a group is. The definition of a group is as follows:

> "Let G be a set together with a binary operation (usually called multiplication) that assigns to each ordered pair (a, b) of elements of G an element in G denoted by ab. We say G is a group under this operation if the following three properties are satisfied.
>
> 1. Associativity: The operation is associative; that is $(ab)c = a(bc)$ for all a, b, c in G.
> 2. Identity. There is an element e (called the identity) in G such that $ae = ea = a$ for all a in G.
> 3. Inverses. For each element a in G, there is an element b in G (called an inverse of a) such that $ab = ba = e$ (Gallian, 2010, pp. 41)."

Thus to show that all orthogonal matrices create a group it must be shown that the three properties listed above associativity, identity, and inverse. Since orthogonal matrices are square matrices with n number of rows and n number of columns, the multiplication of these matrices becomes less complicated.

First associativity must be shown. Meaning that $(AB)C = A(BC)$ for all A, B, C, such that $((AB)C)_{ii} = (A(BC))_{ii}$. By the matrix multiplication definition,

$$((AB)C)_{ii} = \sum_{k=1}^{p}(AB)_{ik}C_{ki} = \sum_{k=1}^{p}\left(\sum_{j=1}^{n}A_{ij}B_{jk}\right)C_{ki} = \sum_{k=1}^{p}\sum_{j=1}^{n}(A_{ij}B_{jk}C_{ki}) =$$

$$\sum_{j=1}^{n}\sum_{k=1}^{p}(A_{ij}B_{jk}C_{ki}) = \sum_{j=1}^{n}A_{ij}\left(\sum_{k=1}^{p}B_{jk}C_{ki}\right) = \sum_{j=1}^{n}A_{ij}(BC)_{ji} = (A(BC))_{ii}.$$ Thus

$(AB)C = A(BC)$, and associativity is shown.

Next is to show the existence of an identity, I, such that $AI = IA = A$. Let A be a matrix with dimension of nxn and elements such that $A_{nxn} = [a]_{ij}; a \in R$ and I be a matrix of dimension nxn and elements such that $\left(I_{nxn}\right)_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$ where i represents the row number and j represents the column number. Hence,

$IA = (0)(A)_{1j} + (0)(A)_{2j} + \ldots + (1)(A)_{ij} + \ldots + (0)(A)_{nj} = (1)(A)_{ij} = (A)_{ij}$. In the same fashion, $AI = (A)_{1j}(0) + (A)_{2j}(0) + \ldots + (A)_{ij}(1) + \ldots (A)_{nj}(0) = (A)_{ij}(1) = (A)_{ij}$. Since

$IA = (A)_{ij} = AI$, an identity exists and it is $\left(I_{nxn}\right)_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$.

Lastly to show the existence of an inverse i.e. $AB = BA = I$. Probably the easiest way to show if an inverse exists of an nxn matrix is to look at its determinant. A theorem from Linear Algebra states "A square matrix A is invertible if and only if the determinant of A is not zero i.e. $\det(A) \neq 0$." Since my set is the set of all orthogonal matrices, based on the definition of orthogonal matrices each matrix is square. Based on the properties of orthogonal matrices the determinant is either 1 or -1 i.e. $\det(A) = \pm 1$. Since the $\det(A) \neq 0$, the matrix is invertible, thus an inverse exists.

Now that all three criteria of the group have been shown I can say that the orthogonal matrices form a group. This group is commonly symbolized as $O(n)$, where n is the size of the matrix or what space to which the matrix belongs.

Special Orthogonal Matrices

Since the orthogonal matrices discussed above can either be used as a rotation or reflection, I will focus on the subset of the matrices that specifically relate to the rotations. Since all orthogonal matrices have $\det(A) = \pm 1$, it is quite easy to separate these matrices into two specific subsets. The first subset will be those matrices with $\det(A) = -1$ and the second will be those matrices with $\det(A) = +1$. The subset with matrices of $\det(A) = -1$ corresponds to the transformation reflections and the subset with matrices of $\det(A) = +1$ corresponds to the transformation rotations. Consider the claim that the subset of matrices of $\det(A) = +1$. This subset of matrices is known as special orthogonal matrices. Since this subset forms entirely from the group $O(n)$ the subset of all orthogonal matrices with $\det(A) = +1$ forms a subgroup called the special orthogonal matrix group or $SO(n)$, where n represents the number of rows and columns of these square matrices (Todd).

Special Orthogonal Matrix Group

Now that the basis of the varying orthogonal matrix groups have been explored it will be important to focus on if the rotational matrices used in the application process fit into this subgroup of special orthogonal matrices $SO(n)$. Before concentrating on the specific matrices used I will start by a more in-depth look at Euler's Rotation Theorem. This is best done by the proof that follows.

"Proof: Our construction is this: if A := $\frac{1}{2}(R - R^T)$ is the skew-symmetric part of the matrix R, then the vector v := (a_{23}, a_{31}, a_{12}) is left fixed by R. This follows from $RAR^T = A$, which is immediate from the definition of A. To see that $RAR^{-1} = RAR^T = A$ is equivalent to Rv = v, we invoke the linear isomorphism $v \rightarrow J_v$ between R^3 and the skew-symmetric 3 x 3 matrices, given explicitly by $J_V = \begin{pmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{pmatrix}$. J_v is just the matrix implementing cross-product with v; i.e., $J_v(w)$:= v x w for all w ∊ IR3. An essential fact about the cross-product operation is its invariance under any such rotation matrix R, that is, R(v x w) = R_v x R_w. In terms of map J, this says $JR_v(R_w) = RJ_v(w)$ or simply $JR_vR = RJ_v$. Rearranging this, we arrive at $JR_v = RJ_vR^{-1}$. (In the language of representation theory, this says that J is an equivalence or "intertwining operator" between the standard representation of SO(3) on R^3 and the adjoint representation on its Lie algebra, SO(3), the skew-adjoint 3 x 3 matrices. Or, in more elementary terms, the matrix for cross-product with a rotated vector is obtained by conjugating (by the rotation) the matrix for cross-product with the un-rotated vector.) It is now immediate that a vector v is fixed by a rotation R if and only if the skew-adjoint matrix J_v is fixed under conjugation by R.

But we are not through yet! What if A = 0? Such orthogonal matrices—the symmetric ones—have measure zero, and correspond to angles of rotation 0 and π. But still they are important, and we have a similar result for them. In this (non-generic) case, $R = R^T$ and so $R^2 = I$. Therefore, $R(I + R) = R + R^2 = I + R$, so the columns of I + R are fixed. Since R is proper (detR = +1), $R \neq -I$, so I + R has a non-zero column that is fixed by R"(Palais, 2007).

Notice that the determinant of the general matrix in the proof above is in fact +1. Meaning that this matrix is in fact a member of $SO(n)$. Recall the rotational matrices used in the application $R_{2c} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ circular rotation in R^2,

$R_{2E} = \begin{bmatrix} \cos(\theta) & -\frac{a}{b}\sin(\theta) \\ \frac{b}{a}\sin(\theta) & \cos(\theta) \end{bmatrix}$ elliptical rotation in R^2, $R_{3C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\sin(t) \\ 0 & \sin(t) & \cos(t) \end{bmatrix}$

circular rotation in R^3, $R_{3E} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\frac{a}{b}\sin(t) \\ 0 & \frac{b}{a}\sin(t) & \cos(t) \end{bmatrix}$ elliptical rotation in R^3. Now with the use of Maple calculating the determinant of each matrix is fairly simple. Thus the determinant of matrix R_{2C}, R_{2E}, R_{3C}, and R_{3E} are each $\cos^2\theta + \sin^2\theta$. Which based on the Pythagorean identity, $\cos^2\theta + \sin^2\theta = 1$ proving that each of these rotational matrices have $\det(A) = +1$ meaning that the rotational matrices used in this application belong to $SO(2)$ and $SO(3)$. To see the complete Maple Programming used refer to Appendix I.

CHAPTER IV

CONCLUSION

In conclusion, this thesis has discussed Euler's Rotation Theorem and how this theorem can be used to help represent the rotation of the solar system with both an elliptical and circular path simultaneously. This is possible because of the use of the rotation matrices that are a part of the special orthogonal matrix group $SO(n)$, specifically $SO(2)$ and $SO(3)$. The theorem allows for the tracking of the center of each sphere, so when the sphere is multiplied by the rotation matrix, the resultant center can be translated as the new position of the resultant sphere. This continual multiplication of each resultant sphere by the respective rotational matrices creates numerous spheres that when animated resembles the earth moving about the sun in an elliptical path as the moon rotates about the earth in a circular path.

REFERENCES

Beckmann, P. (1989). *A History of Pi* (3rd ed.). New York, NY: Dorset Press.

Boyer, C. B. (1968). *A History of Mathematics* (2nd ed.). New York, NY: Wiley.

Bronson, R., & Costa, G. B. (2009). *Matrix Methods Applied Linear Algebra* (3rd ed.). Burlington, MA: Elsevier Inc..

Gallian, J. (2010). Groups. Contemporary Abstract Algebra (7th ed.,). Belmont, CA: Brooks/Cole, Cengage Learning.

Ikenaga, B. (2009, January 1). Properties of Matrix Arithmetic. . Retrieved January 1, 2014, from http://www.millerville.edu/~bikenga/linear-algebra/matrix-propertis/matrix-properties/matrix-properties.html

James, I. (2003). *Remarkable Mathematicians: From Euler to von Neumann.* Cambridge, United Kingdom: University Press, Cambridge, United Kingdom. (Original work published 2002)

Maple 12 Maplesoft, a division of Waterloo Maple Inc., Waterloo, Ontario.

Palais, B., & Palais, R. Euler's fixed point theorem: The axis of a rotation. Journal of Fixed Point Theory and Applications, 2, 215-220.

Shiskowski, K., & Frinkle, K. (2011). Principles of Linear Algebra With Mathematica. Hobocken, NJ: John Wiley & Sons, Inc.

Stakhov, A. P., & Olsen, S. A. (2009). Fibonacci and Lucas Numbers. *The mathematics of harmony from Euclid to contemporary mathematics and computer science* (). Singapore: World Scientific

Todd, R. (n.d.). Orthogonal Matrix. . Retrieved July 25, 2013, from http://mathworld.wolfram.com/OrthogonalMatrix.html

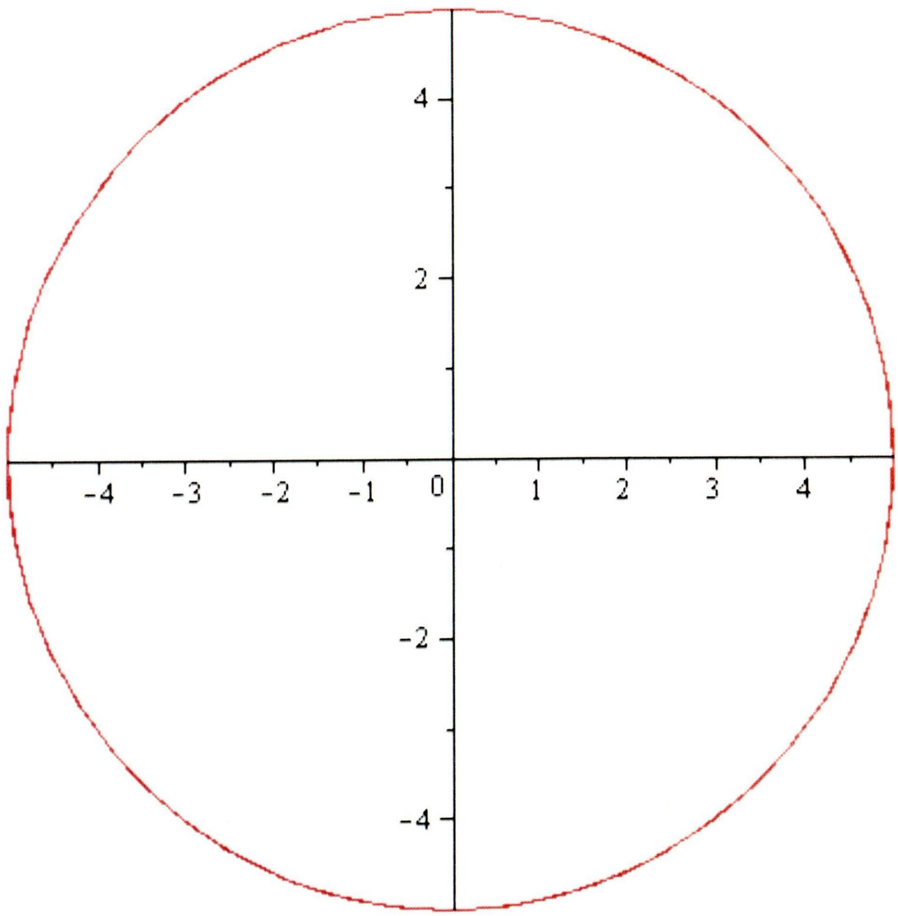
Todd, R. (n.d.). Special Orthogonal Matrix. . Retrieved July 25, 2013, from http://mathworld.wolfram.com/SpecialOrthogonalMatrix.html

APPENDIX A

Maple Coding: Point Rotated in a Circle About the Origin

Point Rotated in a Circle About Origin

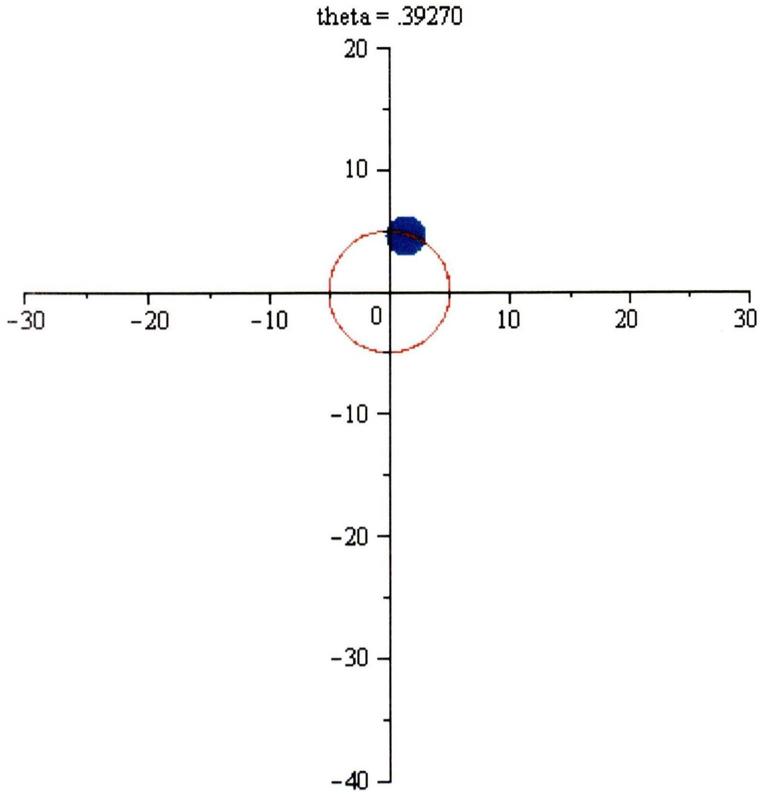
```
> restart:with(linalg): with(plots): with(plottools):
> Q:=matrix(2,1,[3,4]):
> P:=matrix(2,1,[0,0]):
> Rad:=5:
> A:= theta -> matrix([[cos(theta), -sin(theta)],
[sin(theta), cos(theta)]]):
> x:=5*sin(t):
> y:=5*cos(t):
> Cir:=plot([x,y,t=-6..6],color=red):
> display(Cir);
```




```
> T:= theta->evalm(A(theta)&*(Q-P)+P):
> T(theta)[1,1];
```

$$3\cos(\theta) - 4\sin(\theta)$$

```
> Animation:=
animate(pointplot,[[T(theta)[1,1],T(theta)[2,1]]],theta=Pi/
8..16*Pi/8,frames=40,color=blue,symbol=solidcircle,symbolsi
ze=40):
> display({Animation,Cir}, scaling = constrained, view=[-
30..30,-40..20]);
```

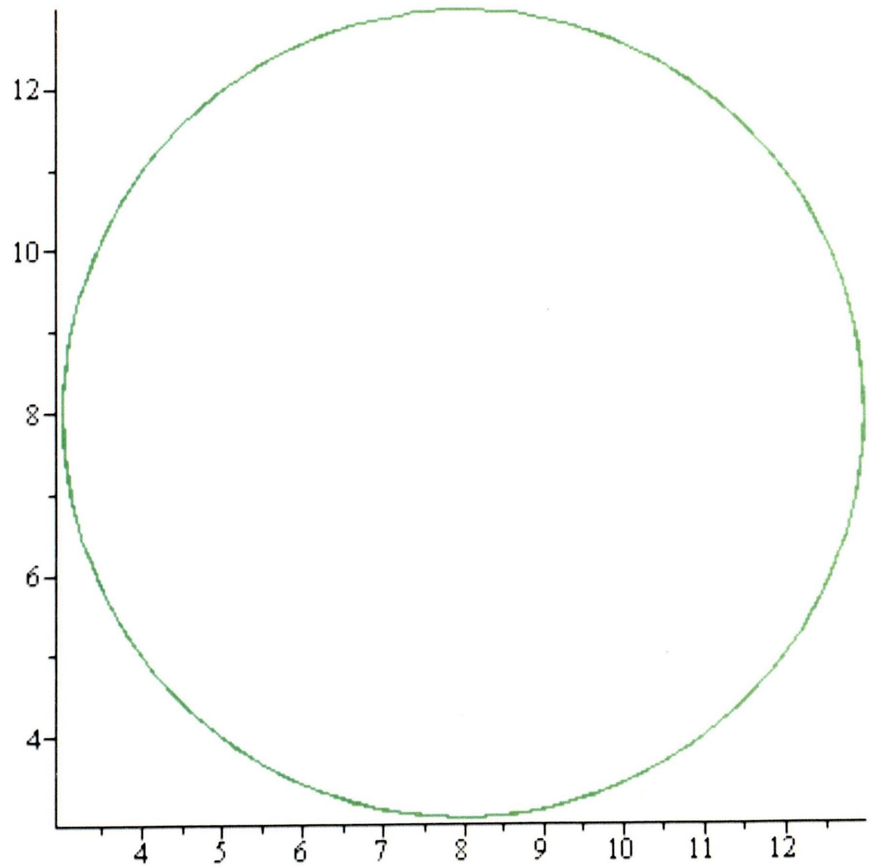


APPENDIX B

Maple Coding: Point Rotated in a Circle About the Point (5,4)

Point Rotated in a Circle About the Point (5,4)

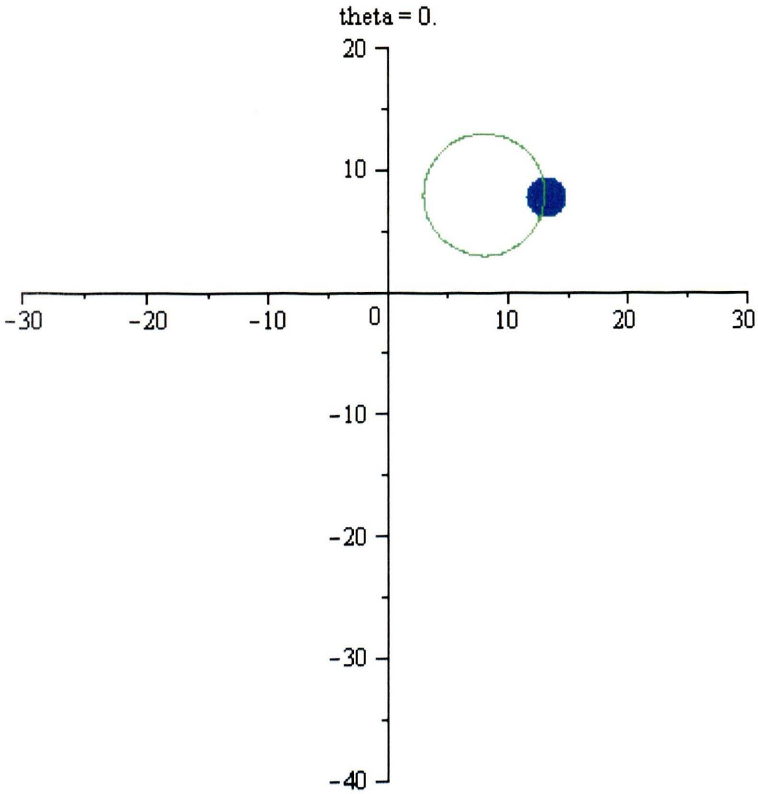
```
> restart:with(linalg): with(plots): with(plottools):
> Q:=matrix(2,1,[13,8]):
> P:=matrix(2,1,[8,8]):
> Rad:=5:
> A:= theta -> matrix([[cos(theta), -sin(theta)],
[sin(theta), cos(theta)]]):
> x:=5*sin(t)+8:
> y:=5*cos(t)+8:
> Cir:=plot([x,y,t=-6..6],color=green):
> display(Cir);
```




```
> T:= theta->evalm(A(theta)&*(Q-P)+P):
> T(theta)[1,1];
```

$$5\cos(\theta) + 8$$

```
> Animation:=
animate(pointplot,[[T(theta)[1,1],T(theta)[2,1]]],theta=0..
2*Pi,frames=30,color=blue,symbol=solidcircle,symbolsize=40)
:
> display({Animation,Cir}, scaling = constrained, view=[-
30..30,-40..20]);
```

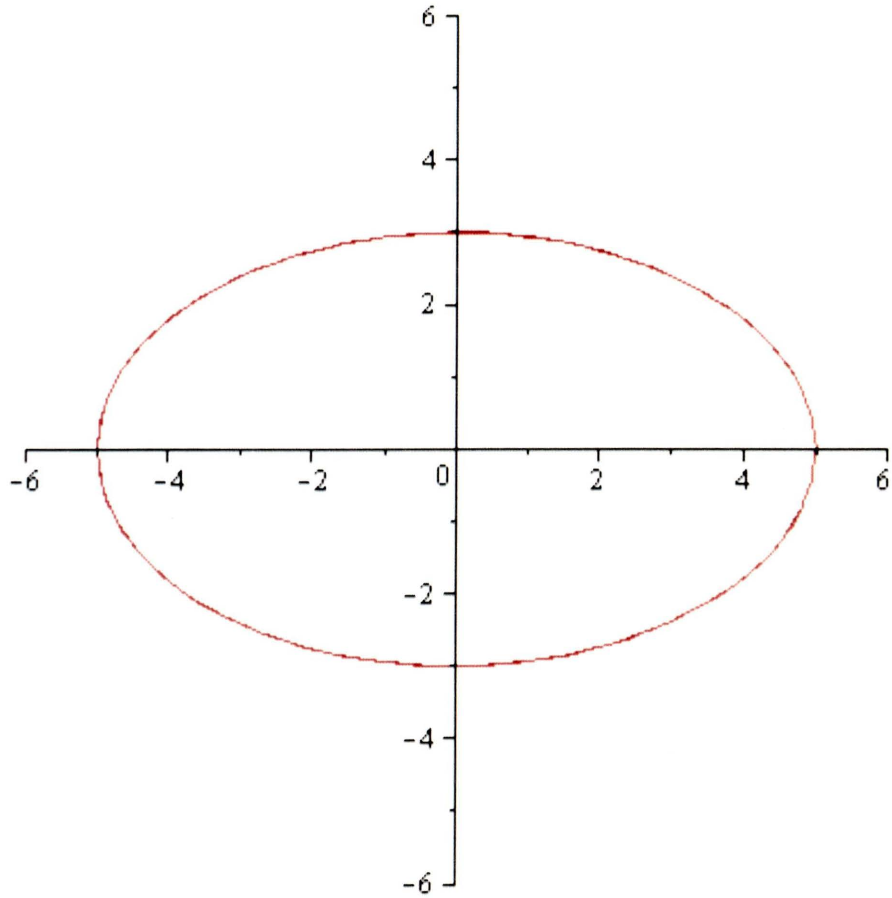


APPENDIX C

Maple Coding: Point Rotated in an Ellipse About the Origin

Point Rotated in an Ellipse About the Origin

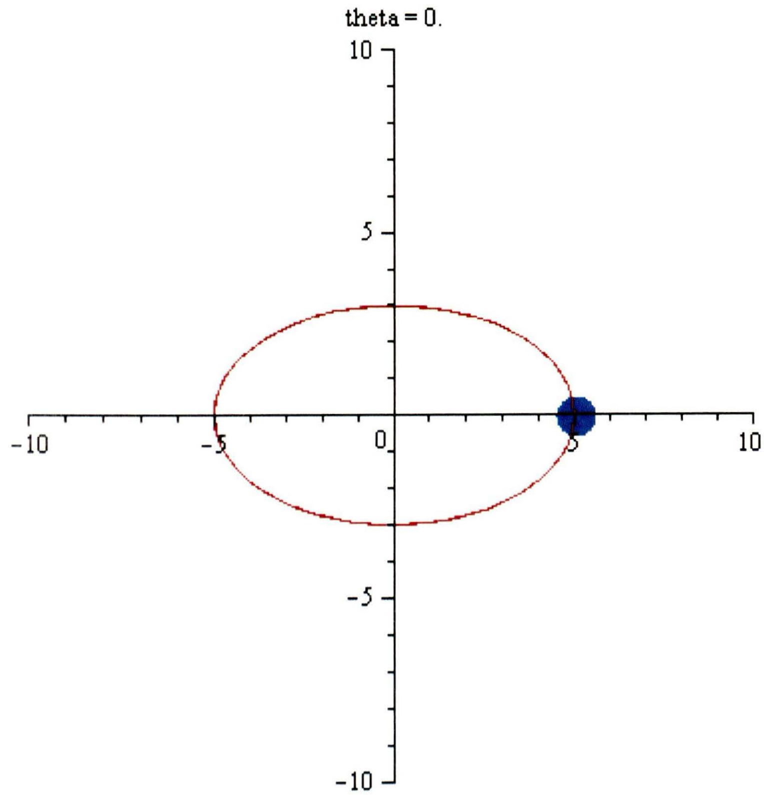
```
> restart:with(linalg): with(plots): with(plottools):
> a:=5:
> b:=3:
> Q:=matrix(2,1,[5,0]):
> P:=matrix(2,1,[0,0]):
> A:= theta -> matrix([[cos(theta), -a/b*sin(theta)],
[b/a*sin(theta), cos(theta)]]):
> x:=5*cos(t):
> y:=3*sin(t):
> Ell:=plot([x,y,t=-6..6],color=orange):
> display(Ell,view=[-6..6,-6..6]);
```




```
> T:= theta->evalm(A(theta)&*(Q-P)+P):> T(theta)[2,1];
```

$$3 \sin(\theta)$$

```
> Animation:= animate(pointplot, [[T(theta)[1,1],
T(theta)[2,1]]], theta = 0..2*Pi, frames = 40, color =
blue, symbol=solidcircle, symbolsize=40):
> display({Animation,Ell}, scaling = constrained, view=[-
10..10,-10..10]);
```

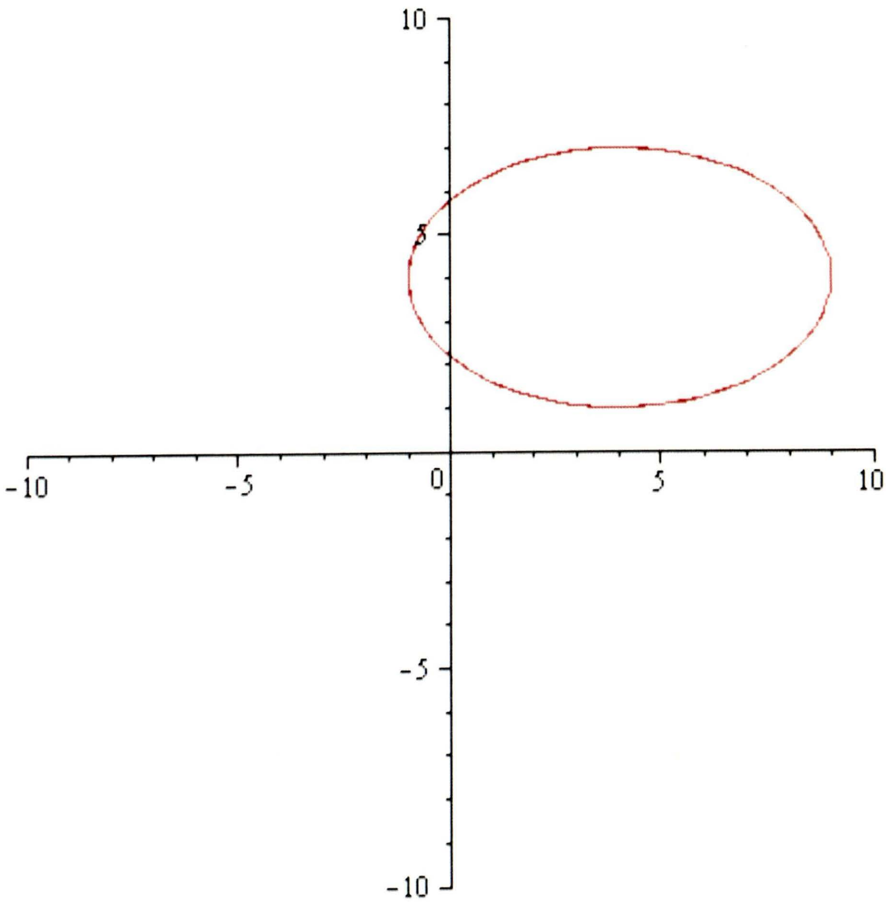


APPENDIX D

Maple Coding: Point Rotated in an Ellipse About the Point (4,4)

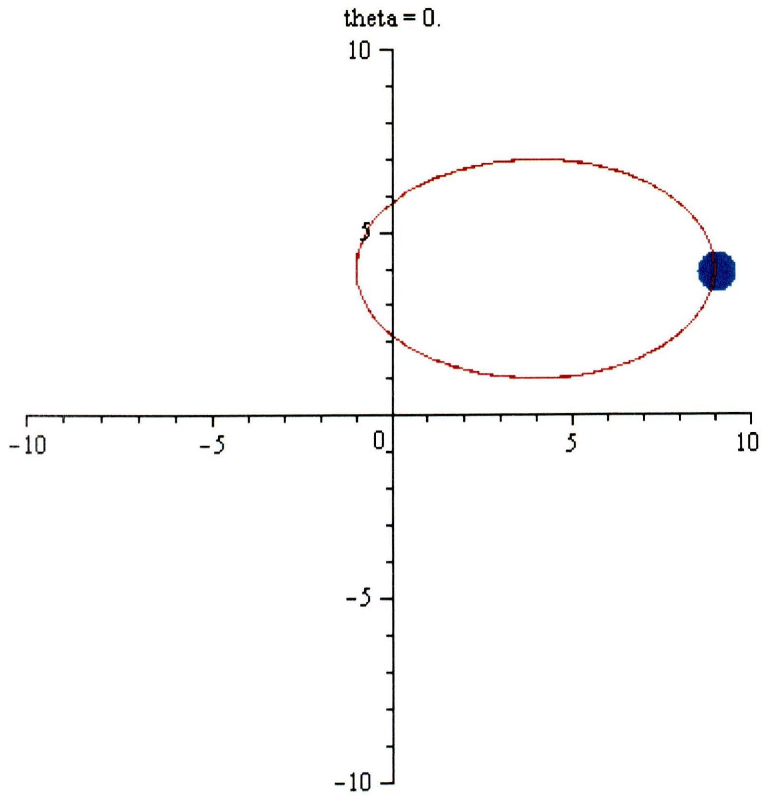
Point Rotated in an Ellipse About the Point (4,4)

```
> restart:with(linalg): with(plots): with(plottools):
> a:=5:
> b:=3:
> Q:=matrix(2,1,[9,4]):
> P:=matrix(2,1,[4,4]):
> A:= theta -> matrix([[cos(theta), -a/b*sin(theta)],
[b/a*sin(theta), cos(theta)]]):
> x:=5*cos(t)+4:
> y:=3*sin(t)+4:
> Ell:=plot([x,y,t=-6..6],color=orange):
> display(Ell,view=[-10..10,-10..10]);
```




```
> T:= theta->evalm(A(theta)&*(Q-P)+P):>
T(theta)[1,1]:
```

```
> Animation:= animate(pointplot, [[T(theta)[1,1],
T(theta)[2,1]]], theta = 0..2*Pi, frames = 40, color =
blue, symbol=solidcircle, symbolsize=40):
> display({Animation,Ell}, scaling = constrained, view=[-
10..10,-10..10]);
```

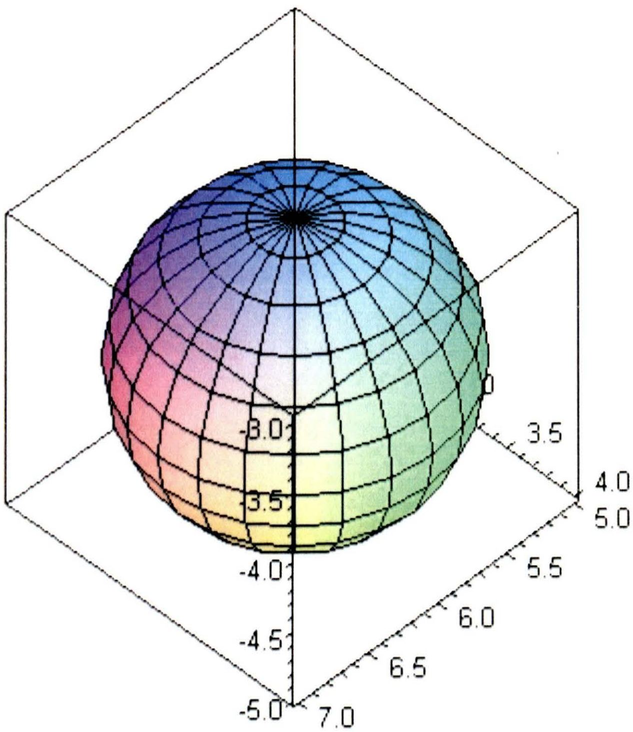


APPENDIX E

Maple Coding: Sphere Rotated in a Circle Around the X-Axis

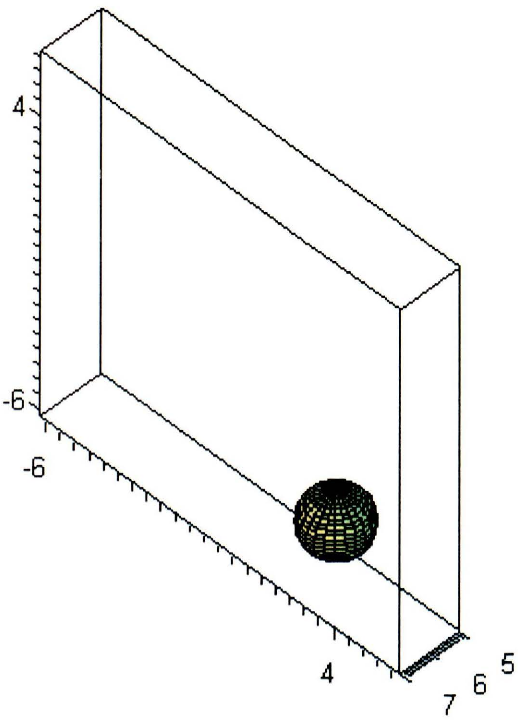
Sphere Rotated in a Circle Around the X-Axis

```
> restart:
> with(linalg): with(plots): with(plottools):
>
s:=[1*sin(phi)*sin(theta)+6,1*sin(phi)*cos(theta)+3,1*cos(p
hi)-4]:
> t:=array(s):
> sp:=plot3d(s, phi=-Pi..Pi, theta=-Pi..Pi):
> display3d(sp,axes = boxed, scaling = constrained);
```




```
>
p:=array([1,0,0]):
> c:=evalm(dotprod(p,t)/dotprod(p,p)*p):
> r:=evalm(norm(t-c,2)):
> U:=evalm((t-c)/r):
> V:=evalm(crossprod(c,U)/norm(crossprod(c,U),2)):
> sprot:=A-
>evalf(convert(evalm(c+r*cos(A)*U+r*sin(A)*V),list)):
```

```
> movingsp:=animate3d(sprot(A), phi = 0..Pi, theta =
0..2*Pi, A = 0..6,frames=20):
>
display3d(movingsp,axes=boxed,scaling=constrained,insequenc
e=true);
```

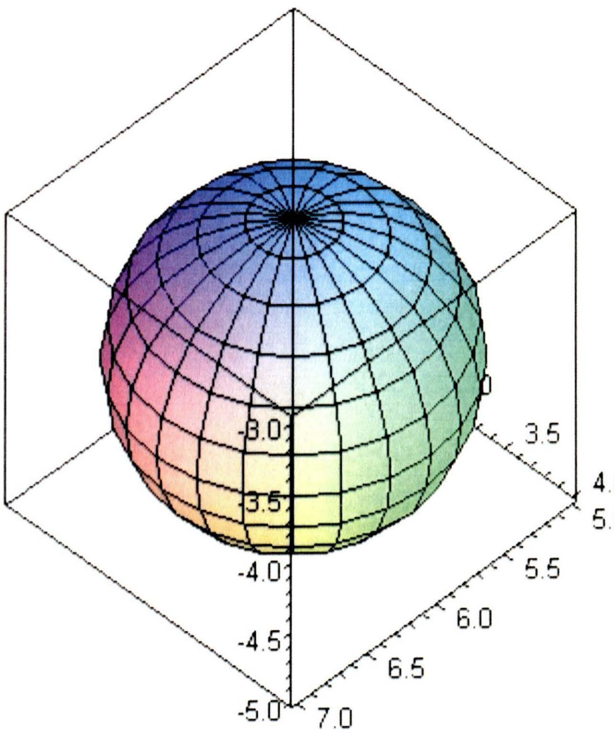


APPENDIX F

Maple Coding: Sphere Rotated in a Circle About the Point (3,4,5)

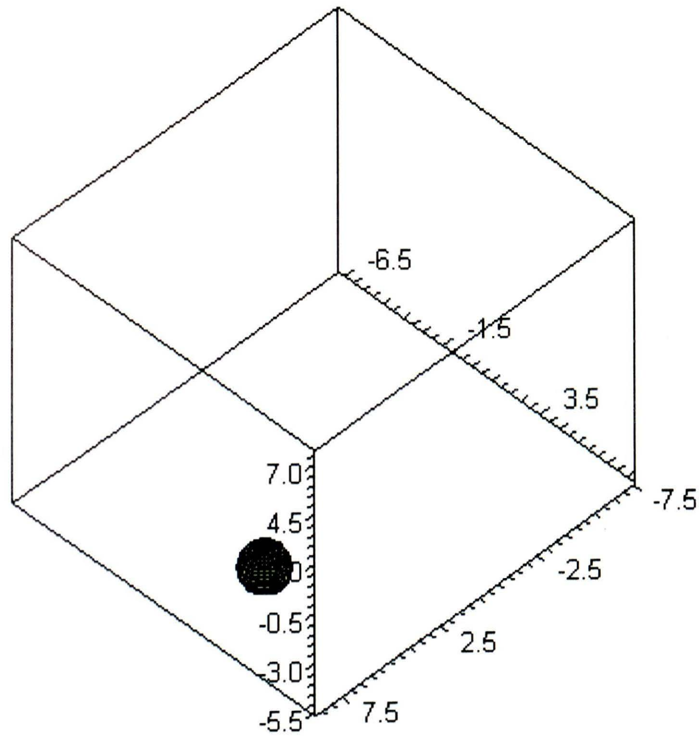
Sphere Rotated in a Circle Around the Point (3,4,5)

```
> restart:
> with(linalg): with(plots): with(plottools):
>
s:=[1*sin(phi)*sin(theta)+6,1*sin(phi)*cos(theta)+3,1*cos(p
hi)-4]:
> t:=array(s):
> sp:=plot3d(s, phi=-Pi..Pi, theta=-Pi..Pi):
> display3d(sp,axes = boxed, scaling = constrained);
```




```
>
p:=array([3,4,5]):
> c:=evalm(dotprod(p,t)/dotprod(p,p)*p):
> r:=evalm(norm(t-c,2)):
> U:=evalm((t-c)/r):
> V:=evalm(crossprod(c,U)/norm(crossprod(c,U),2)):
> sprot:=A-
>evalf(convert(evalm(c+r*cos(A)*U+r*sin(A)*V),list)):
```

```
> movingsp:=animate3d(sprot(A), phi = 0..Pi, theta =
0..2*Pi, A = 0..6, frames=20):
>
display3d(movingsp,axes=boxed,scaling=constrained,insequenc
e=true);
```

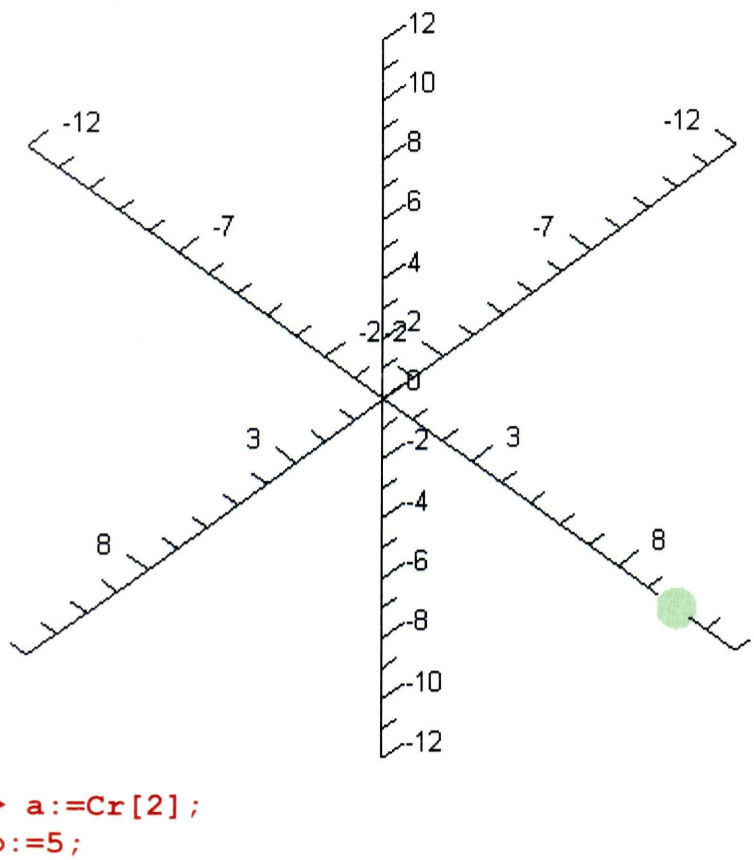


APPENDIX G

Maple Coding: Sphere Rotated About the Ellipse $\frac{x^2}{100} + \frac{y^2}{25} = 1$

Sphere Rotated About the Ellipse $\frac{x^2}{100}+\frac{y^2}{25}=1$

```
> restart:
> with(linalg): with(plots): with(plottools):
> assume(phi,real):assume(theta,real):
> Cr:=[0,10,0]:
> rad:=1/2:
>
s:=[Cr[1]+rad*sin(phi)*sin(theta),Cr[2]+rad*sin(phi)*cos(th
eta),Cr[3]+rad*cos(phi)]:
> sp:=plot3d(s, phi=-Pi..Pi, theta=-Pi..Pi):
> display3d(sp,axes = boxed, scaling = constrained, view=[-
12..12,-12..12,-12..12], style=patchnogrid, axes=normal,
scaling=constrained);
```




```
> a:=Cr[2];
b:=5;
```

$a := 10 b := 5$

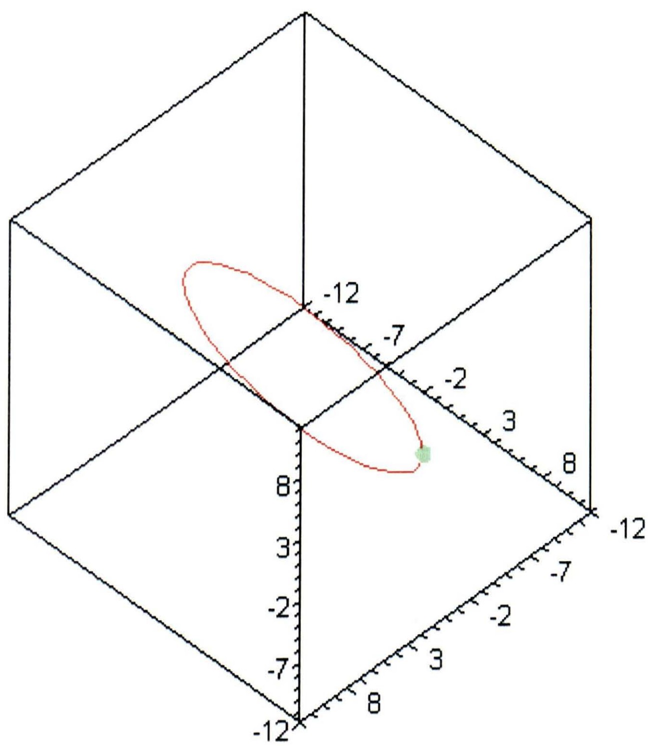
```
> A:= theta -> matrix(3,3,[1,0,0,0,cos(theta),-
a/b*sin(theta),0,b/a*sin(theta),cos(theta)]):
> A(t);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -2\sin(t) \\ 0 & \frac{1}{2}\sin(t) & \cos(t) \end{bmatrix}$$

```
> Q:= t -> evalm(A(t)&*Cr):
> Q(7*Pi/8);
```

$$\left[0 \quad -10\cos\left(\frac{1}{8}\pi\right) \quad 5\sin\left(\frac{1}{8}\pi\right) \right]$$

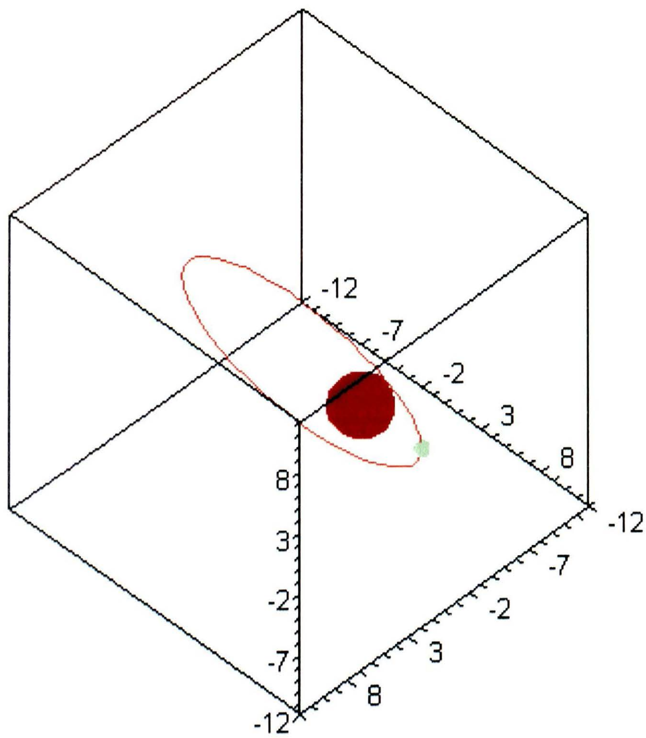
```
> EllipsePlot:=spacecurve([0,a*cos(t),b*sin(t)],t=0..2*Pi,
thickness=1,color=RED):
> SpherePlots:=
seq(plot3d([Q(k*Pi/6.)[1]+rad*sin(phi)*sin(theta),Q(k*Pi/6.
)[2]+rad*sin(phi)*cos(theta),Q(k*Pi/6.)[3]+rad*cos(phi)],
phi=-Pi..Pi, theta=-Pi..Pi), k=0..12):
> display3d([EllipsePlot,display3d(SpherePlots,insequence =
true)], axes = boxed, scaling= constrained, view=[-12..12,-
12..12,-12..12], style=patchnogrid);
```



```
> Foci:=[0,5,0];
```

$$Foci := [0, 5, 0]$$

```
> SunSphere := sphere(Foci, 2, color=orange):
> display3d([EllipsePlot,
SunSphere,display3d(SpherePlots,insequence = true)], axes =
boxed, scaling= constrained, view=[-12..12,-12..12,-
12..12], style=patchnogrid);
```



APPENDIX H

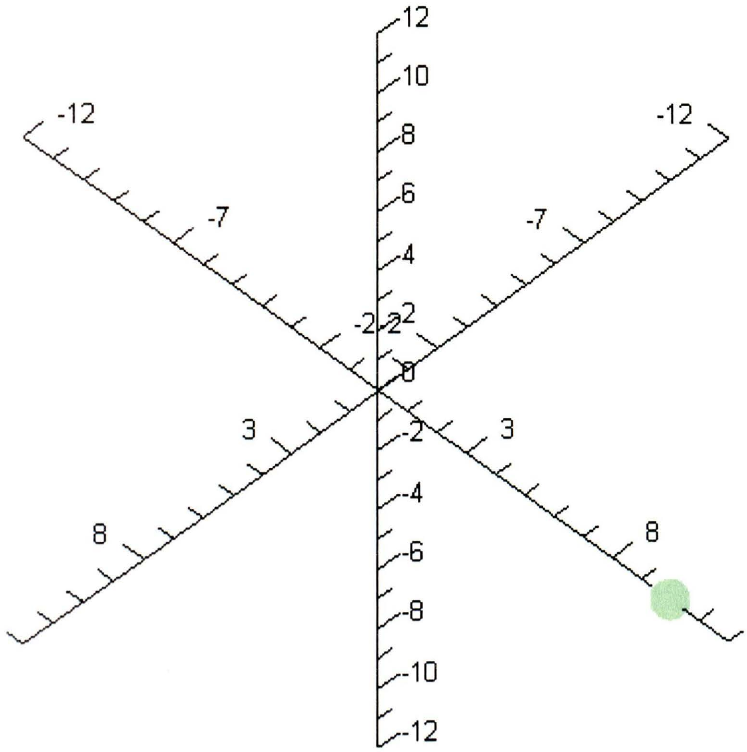
Maple Coding: Rotation of the Earth in an Elliptical Path While the Moon Rotates in a Spherical Path Around the Earth

Rotation of the Earth in an Elliptical Path While the Moon Rotates in a Spherical Path Around the Earth.

```
> restart:
> with(linalg): with(plots): with(plottools):
> assume(phi,real):assume(theta,real):
> CrE:=[0,10,0]:
CrM:=[0,11,0]:
> radE:=1/2:
radM:=1/4:
>
earth:=[CrE[1]+radE*sin(phi)*sin(theta),CrE[2]+radE*sin(phi
)*cos(theta),CrE[3]+radE*cos(phi)];
moon:=[CrM[1]+radM*sin(phi)*sin(theta),CrM[2]+radM*sin(phi)
*cos(theta),CrM[3]+radM*cos(phi)]:
```

$$earth := \left[\frac{1}{2}\sin(\phi\sim)\sin(\theta\sim),\ 10+\frac{1}{2}\sin(\phi\sim)\cos(\theta\sim),\ \frac{1}{2}\cos(\phi\sim)\right]$$

```
> earthp:=plot3d(earth, phi=-Pi..Pi, theta=-Pi..Pi):
moonp:=plot3d(moon, phi=-Pi..Pi, theta=-Pi..Pi):
> display3d(earthp,axes = boxed, scaling = constrained,
view=[-12..12,-12..12,-12..12], style=patchnogrid,
axes=normal, scaling=constrained);
```




```
> a:=CrE[2]:
b:=7:
> A:= theta -> matrix(3,3,[1,0,0,0,cos(theta),-
a/b*sin(theta),0,b/a*sin(theta),cos(theta)]):
B:= theta -> matrix(3,3,[1,0,0,0,cos(theta),-
sin(theta),0,sin(theta),cos(theta)]):
> A(t);
B(t);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\frac{10}{7}\sin(t) \\ 0 & \frac{7}{10}\sin(t) & \cos(t) \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(t) & -\sin(t) \\ 0 & \sin(t) & \cos(t) \end{bmatrix}$$

```
> Q:= t -> evalm(A(t)&*CrE):
R:= t -> evalm(B(t)&*evalm(CrM-CrE)):
> Q(7*Pi/8);
R(7*Pi/8);
```

$$\left[\begin{array}{ccc} 0 & -10\cos\left(\frac{1}{8}\pi\right) & 7\sin\left(\frac{1}{8}\pi\right) \end{array}\right]$$

$$\left[\begin{array}{ccc} 0 & -\cos\left(\frac{1}{8}\pi\right) & \sin\left(\frac{1}{8}\pi\right) \end{array}\right]$$

```
> EllipsePlot:=spacecurve([0,a*cos(t),b*sin(t)],t=0..2*Pi,
thickness=1,color="DimGray"):
SunPlot:=sphere([0, -5, 0], 1, color="OrangeRed",
style=patchnogrid):
> EMrad:=norm(evalm(CrM-CrE),frobenius);
```

$$EMrad := 1$$

```
> Earths[1]:=
[Q(0)[1]+radE*sin(phi)*sin(theta),Q(0)[2]+radE*sin(phi)*cos
(theta),Q(0)[3]+radE*cos(phi)];
Moons[1]:=
evalm([R(0)[1]+radM*sin(phi)*sin(theta),R(0)[2]+radM*sin(ph
i)*cos(theta),R(0)[3]+radM*cos(phi)]+CrE);
MoonOrbit[1]:=[Q(0)[1],Q(0)[2]+EMrad*cos(t),Q(0)[3]+EMrad*s
in(t)];
```

$$Earths_1 := \left[\frac{1}{2}\sin(\phi\sim)\sin(\theta\sim), 10+\frac{1}{2}\sin(\phi\sim)\cos(\theta\sim), \frac{1}{2}\cos(\phi\sim)\right]$$

$$Moons_1 := \left[\begin{array}{ccc} \frac{1}{4}\sin(\phi\sim)\sin(\theta\sim) & 11+\frac{1}{4}\sin(\phi\sim)\cos(\theta\sim) & \frac{1}{4}\cos(\phi\sim) \end{array}\right]$$

$$MoonOrbit_1 := [0, 10+\cos(t), \sin(t)]$$

```
> n:=100:
> for k from 1 to n do
 Earths[k+1]:=
[Q(2.*k*Pi/n)[1]+radE*sin(phi)*sin(theta),Q(2.*k*Pi/n)[2]+r
adE*sin(phi)*cos(theta),Q(2.*k*Pi/n)[3]+radE*cos(phi)]:
 Moons[k+1]:=
([Q(2.*k*Pi/n)[1],Q(2.*k*Pi/n)[2],Q(2.*k*Pi/n)[3]]+[R(18.*k
*Pi/n)[1]+radM*sin(phi)*sin(theta),R(18.*k*Pi/n)[2]+radM*si
n(phi)*cos(theta),R(18.*k*Pi/n)[3]+radM*cos(phi)]):

MoonOrbit[k+1]:=[Q(2.*k*Pi/n)[1],Q(2.*k*Pi/n)[2]+EMrad*cos(
t),Q(2.*k*Pi/n)[3]+EMrad*sin(t)]:
```

```
end do:
> for k from 1 to (n+1) do
 EarthPlots[k]:= plot3d(Earths[k],phi=-Pi..Pi, theta=-
Pi..Pi,color=blue):
 MoonPlots[k]:= plot3d(Moons[k],phi=-Pi..Pi, theta=-
Pi..Pi,color=black):
 MoonOrbitPlots[k]:=
spacecurve(MoonOrbit[k],t=0..2*Pi,color="DimGray"):
end do:
>
display([SunPlot,EllipsePlot,display3d([seq(display3d([Eart
hPlots[k],MoonPlots[k],MoonOrbitPlots[k]]),k=1..n+1)],inseq
uence=true)], axes = none, scaling= constrained, view=[-
5..5,-15..15,-15..15], style=patchnogrid, labels=[x,y,z]);
```