ROTATIONS AND ORIENTATIONS IN R^3

Paul Bouthellier
Department of Computer Science and Mathematics
University of Pittsburgh-Titusville
Titusville, PA 16354
pbouthe@pitt.edu

In this paper paths along which objects travel will be created by parametric curves in R^2 and R^3. The orientation of the objects along the paths will then be created by rotation matrices and quaternions. Problems with the orientation of objects along the path, such as Gimbal lock and wrap-around infinity will also be looked at. These concepts will be illustrated via computer images created using the 3D graphics packages Studio 3D Max, Carrara, Poser and the programming language Python.

The orientation of objects along paths will be done by converting the tangent and normal vectors along paths to arctangents and spherical angles (θ,φ).

Two well-known problems associated with orientations along paths will then be illustrated: Gimbal lock and wrap-around infinities. Gimbal lock occurs when a rotation about one axis causes a loss of the ability to rotate about one of the three coordinate axes. Wrap-around infinities occurs where the x, y, or z derivatives along the path become undefined. As orientations are based on the inverse tangent, an object can perform a rotation from $-\pi/2$ to $\pi/2$ or $\pi/2$ to $-\pi/2$ (plus some phase angle) at such points. A problem where Gimbal lock and wrap-around infinity occurred in practice will be illustrated.

Paths

Figure 1-A stylized vector

Figure 1 is a still from an animation of an object flying along a figure 8 path in R^2. Paths in R^2 and R^3 are generally created by parametric equations of the form:

$$f(t)=\langle x(t),y(t),z(t)\rangle$$

where in simple cases x(t), y(t), z(t) are often either polynomials, polar equations, trigonometric functions or Bezier curves.

In situations where the flight of an object needs to be precisely modeled x(t), y(t), z(t) may involve factors such as:

- The mass of the object and the distribution of said mass
- The cross section of the object
- The objects rotation (for lift)
- An atmospheric model
- The effects of wind
- The rotation of the Earth (Recall the Paris gun)
- Gravity
- The object's rotation and orientation along its path (roll, yaw, pitch)
- The object's tendency to go into Gimbal lock

Creating such paths can then involve:

- Time-varying systems of differential equations
- Rotation matrices
- Quaternions
- Every approximation technique ever invented.

Orientations Along Paths

In R^2 objects move along a path given by f(t)=<x(t),y(t)>. A typical orientation is given by

$$\text{Orientation}(t)=\text{atan}(\text{Slope}(f'(t)))$$

A problem will this method is when we hit a time t where the slope is undefined. If the slope changes from $-\infty$ to $+\infty$ or vice versa the orientation of the object can suddenly switch by 180°-this is known as a wrap-around infinity. We shall discuss this in more detail below.

In R^3 objects move along a path given by f(t)=<x(t),y(t),z(t)>. A typical orientation is done is two parts:

1. Compute the tangent vector $f'(t)=\langle x'(f),y'(t),z'(t)\rangle$
2. Translate this vector into spherical angles (θ,φ)

The spherical angles (θ,φ) are then used in matrices to control the orientations along the path.

A problem will this method is when that of Gimbal lock. Gimbal lock causes us to lose a degree of freedom when trying to perform rotations. We shall discuss this in detail below.

Problems with Orientations Along Paths

- Wrap-Around Infinities
- Gimbal Lock
- Interpolation Problems

Before beginning, I wish to state what I consider the most important part of this paper: Before trying to orient objects along paths in either mathematical, physical, engineering, or computer applications-do a detailed study of Gimbal lock, Euler angles, quaternions, matrix rotations, interpolation with matrices and quaternions, and wrap-around infinities. These problems can cause havoc to the most well thought out design otherwise. They must be dealt with.

Wrap-Around Infinites

In R^2 we saw that Orientation(t)=atan(Slope(f′(t))). Consider a simple path given as follows:

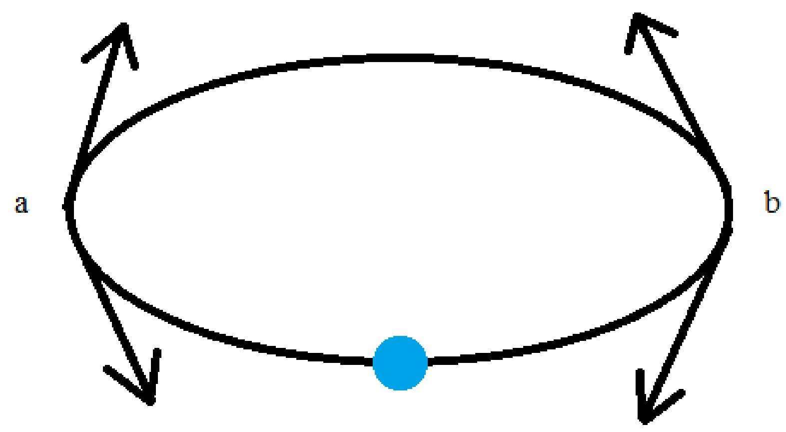

Figure 2

An obvious problem exists at points a and b along the path. As our object (a planet in this case) reaches points a and b –considering a counterclockwise rotation-the slope of the path changes from negative to positive (-∞ to +∞) to positive and positive to negative (+∞ to -∞) respectively.

Hence the name wrap-around infinity. This causes our poor planet to change its orientation from –π/2 to π/2 at point a and b from π/2 to -π/2 at point b.

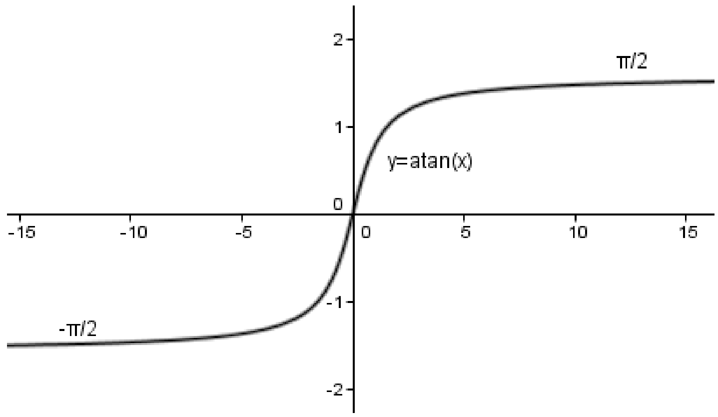

Figure 3

An example-which occurred in actual practice is illustrated below.

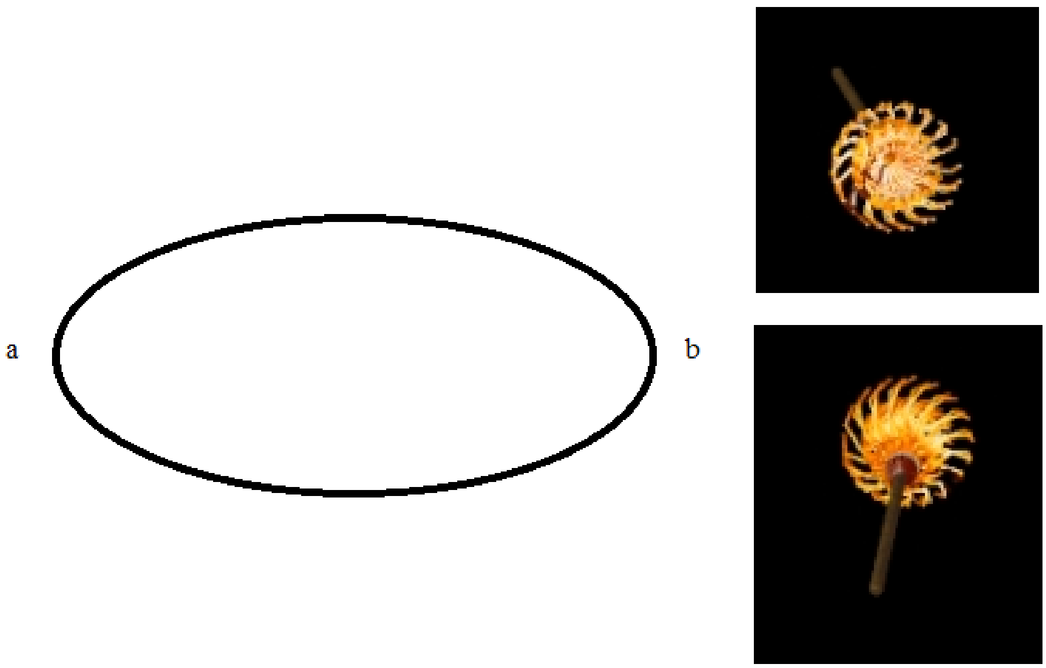

Figure 4

To avoid the problem the algorithm often has to be rewritten at such points to avoid this problem.

Gimbal Lock:

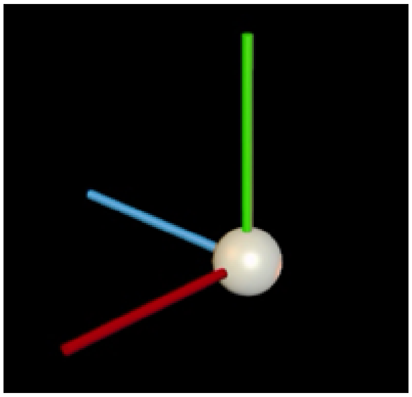

Figure 5a

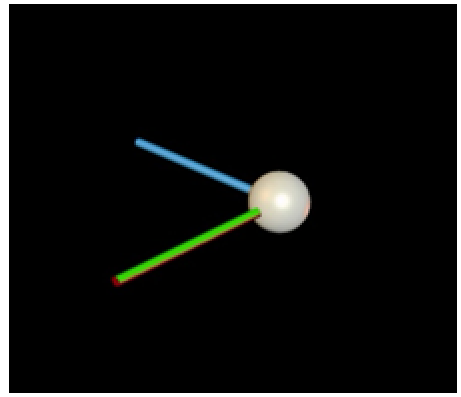

Figure 5b

Gimbal lock is a loss of one degree of freedom when doing rotations. (Look at any pictures of the command modules from the Apollo program-they had an indicator warning is the ship was about to enter Gimbal lock-meaning they could only rotate the ship about 2-axis-beyond major trouble.)

In many applications involving rotations in three-space, rotations are done as follows:

- An order of rotation is specified: for example XZY. This means that the rotation about the X-axis is done first, then rotation about the Z-axis is done second, and finally the rotation about the Y-axis is done. We shall consider an XZY rotation sequence order in the following. In the above figures red=x-axis, green=y-axis, blue=z-axis.
- When the rotation about the X-axis is done, the Z and Y-axes are rotated as well.
- When the second axis is rotated, Z in this case, only the third axis Y is rotated with it. If this second axis is rotated either 90 or -90 degrees the Y-axis is rotated onto the X-axis causing a loss of a degree of freedom in terms of rotations. This is Gimbal lock.

Even if the second rotation is near -90 or 90 degrees the rotations can still become unstable near those angles. The standard way to minimize the problems of Gimbal lock is as follows:

- Decide how much a given object needs to rotate about the various axes.
- Choose a rotation order which assigns the second rotation axis to this angle which requires the minimum amount of rotation.
- Hope this solve any problems.
- Use quaternions.

Quaternions basically define a rotation, which is specified in a term called q_0 about a vector $q=\langle q_1,q_2,q_3\rangle$. (Four terms-hence quaternion). Instead of being a sequence of rotations, which is subject to Gimbal lock, quaternions are a single rotation about a vector which does not suffer from Gimbal lock.

The down side to quaternions is that most people do not find them very intuitive. However, given they are used extensively in areas such as computer graphics and modeling, they should, in this authors opinion, be included in courses such as linear algebra. For far more details about quaternions see [1], [2]:

An example-which occurred in actual practice is illustrated below. Our object was supposed to rotate about its vertical axis as it rotated about its path. However, as it was in Gimbal lock it did no such rotation. Changing the order of rotations will often fix such problems.

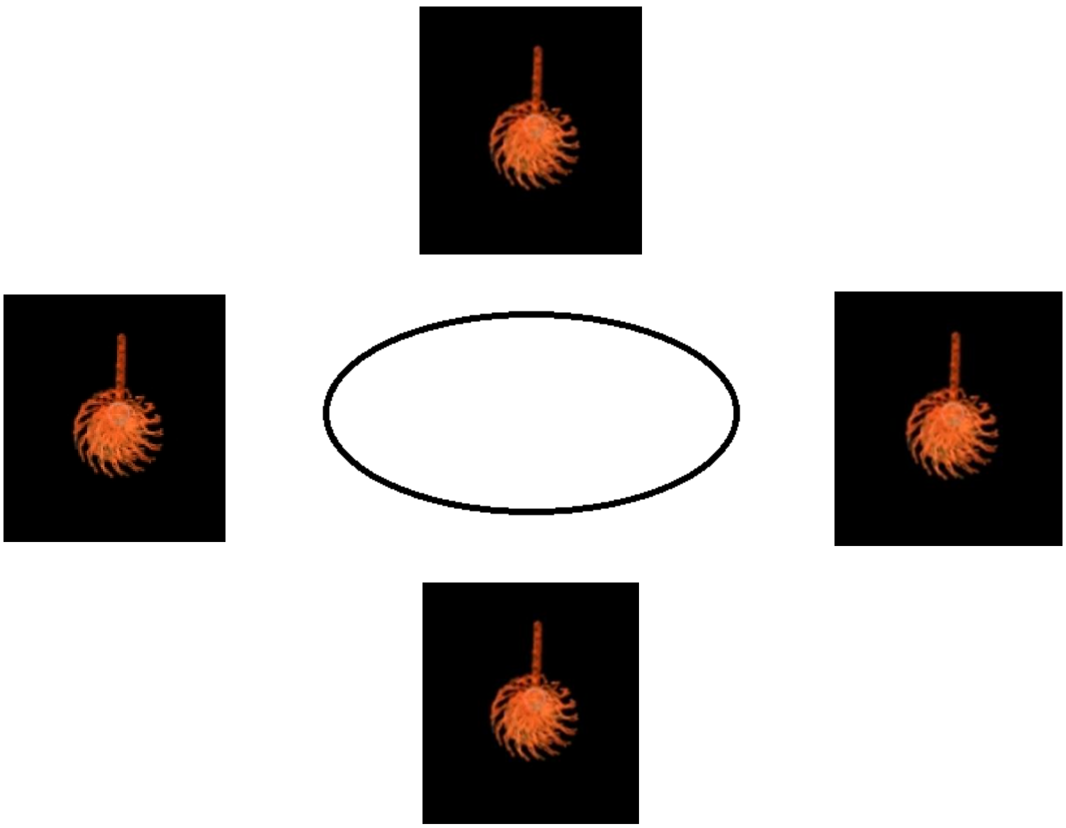

Figure 6

Interpolation Problems:

When orienting objects along paths, a typical way to approach the problem is to orient an object at a finite set of points t_0, t_1, t_2,…,tn.

As an example, consider the XZY rotation sequence considered in the Gimbal lock section above. One could define the starting orientation and the terminal orientations as follows:

Starting Orientation	Terminal Orientation

$X=\theta x_s$	$X=\theta x_t$
$Z=\theta z_s$	$Z=\theta z_t$
$Y=\theta y_s$	$Y=\theta y_t$

As time progresses from the starting time s to the terminal t time the angles change from the starting values to the terminal values along some defined path. However, if the second angle, in this case θy, takes on (or gets close to) the values of $-\pi/2$ or $\pi/2$ the system will approach Gimbal lock and the rotation and hence orientation of the system may become unstable. Hence, as stated above, the best way to avoid this problem is to try to order the sequence of rotations such that the second rotation does not pass through either of the values $-\pi/2$ or $\pi/2$.

References:

[1] Andrew Hanson, "*Visualizing Quaternions*" San Francisco, CA: Morgan Kaufmann, 2006.

[2] Jack B. Kuipers, "*Quaternions and Rotation Sequences*" Princeton, NJ: Princeton University Press, 2002.