Maths - AxisAngle to Quaternion

- Axis-Angle to Quaternion Calculator.

Prerequisites

Definition of terms:

- Axis Angle
- Quaternions

Equations

```
qx = ax * sin(angle/2)
qy = ay * sin(angle/2)
qz = az * sin(angle/2)
qw = cos(angle/2)
```

where:

Standards

There are a lot of choices we need to make in mathematics, for example,

- Left or right handed coordinate systems.
- Vector shown as row or column.
- Matrix order.
- Direction of x,y and z coordinates.
- Euler angle order
- Direction of positive angles
- Choice of basis for bivectors
- Etc. etc.

- the axis is normalised so: ax*ax + ay*ay + az*az = 1
- the quaternion is also normalised so cos(angle/2)^2 + ax*ax * sin(angle/2)^2 + ay*ay * sin(angle/2)^2+ az*az * sin(angle/2)^2 = 1

Code

Java code to do conversion:

```
// assumes axis is already normalised
public void set(AxisAngle4d a1) {
  double s = Math.sin(a1.angle/2);
  x = a1.x * s;
  y = a1.y * s;
  z = a1.z * s;
  w = Math.cos(a1.angle/2);
}
```

Derivation of Equations

see quaternion representation of rotations.

This can be proved as follows:

from trig formula we get

cos(angle/2)^2 + sin(angle/2)^2 = 1

multiplying th sine part by 1 = ax*ax + ay*ay + az*az will have no effect so we can write:

cos(angle/2)^2 + (ax*ax+ ay*ay + az*az) * sin(angle/2)^2 = 1

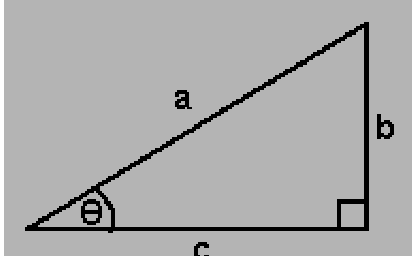
expanding out gives:

cos(angle/2)^2 + ax*ax * sin(angle/2)^2 + ay*ay * sin(angle/2)^2+ az*az * sin(angle/2)^2 = 1

This shows that the quaternion is normalised since it is in the form:

A lot of these choices are arbitrary as long as we are consistent about it, different authors tend to make different choices and this leads to a lot of confusion. Where standards exist I have tried to follow them (for example x3d and MathML) otherwise I have at least tried to be consistent across the site. I have documented the choices I have made on this page.

Trigonometry Functions



The trigometric functions, such as sin, cos and tan, represent how the ratio of these lengths depends on

$qw^2 + qx^2 + qy^2 + qz^2 = 1$

and if we take the square root of each of these terms we get the parts of the quaternion:

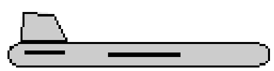
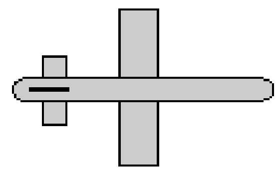
qw = cos(angle/2)
qx = ax * sin(angle/2)
qy = ay * sin(angle/2)
qz = az * sin(angle/2)

That last part was not exactly a rigourous proof, but we can easily check that it is correct by checking rotations about each axis seperately as is done here.

Issues

Most maths libraries use radians instead of degrees (apart from OpenGL).

Example

we take the 90 degree rotation from this:  to this: 

As shown here the axis angle for this rotation is:

angle = 90 degrees
axis = 1,0,0

So using the above result:

cos(45 degrees) = 0.7071

sin(45 degrees) = 0.7071

qx= 0.7071

the angle θ and we can define the following functions for each of the ratios:

sine function	sin(θ) = b / a
cosine function	cos(θ) = c / a
tangent function	tan(θ) = b / c
cosecant	cosec(θ) = a / b
secant	sec(θ) = a / c
cotangent	cot(θ) = c / b

This page explains this.

qy = 0

qz = 0

qw = 0.7071

this gives the quaternion (0.7071+ i 0.7071) which agrees with the result here

Angle Calculator and Further examples

I have put a java applet here which allows the values to be entered and the converted values shown along with a graphical representation of the orientation.

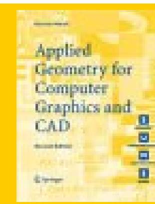
Also further examples in 90 degree steps here

metadata block

see also:	• other convertions • Euler Angles • Matrix • Rotations
Correspondence about this page	• Alexis • help for new programmer

Book Shop - Further reading.

Where I can, I have put links to Amazon for books that are relevant to the subject, click on the appropriate country flag to get more details of the book or to buy it from them.



Applied Geometry for Computer Graphics...

This site may have errors. Don't use for critical systems.