

# Euler To Quaternion

short description	Takes 3 rotation angles (x/y/z rot) and converts them to x/y/z/w quaternions
ports	Bank (X) [control/numeric]
	Heading (Y) [control/numeric]
	Altitude (Z) [control/numeric]
	x Quaternion [control/numeric]
	y Quaternion [control/numeric]
	z Quaternion [control/numeric]
	w Quaternion [control/numeric]

## used in example

- [Moving Screens](#)

## Manual

This is not in the manual.

However, there is an extensive explanation in the a.m. example:

When creating this project I ran into a strange problem: x, y, and z rotation are no independent from each other. I described it like this:

... in AI, the x and y rotation of a screen fixture refer to the world's x and y axis. However, z rotation refers to the screen's z axis. This leads to the situation - with x\_rot = -90 - that y\_rot and z\_rot do exactly the same, but the screen cannot be tilted sideways (what z\_rotation does).

Dave Green was kind enough to give a thorough explanation of this:

You have discovered one of the draw backs of the Euler angle system we use in to rotate screens in Ai. Gimbal Lock. There is a short video explaining the problem here: <https://www.youtube.com/watch?v=zc8b2Jo7mno>. The advantage to using the Euler system for angles is that everybody understands it. But the trade off is that it has limitations in the form of gimbal lock.

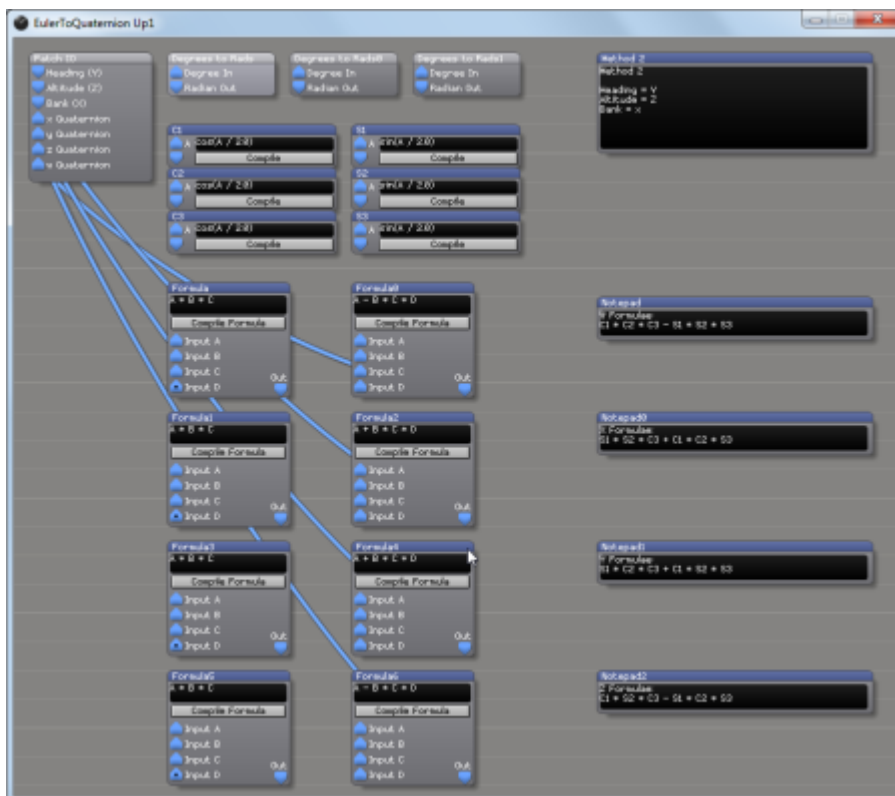
So you are probably thinking what can I do to resolve this problem? Well the simplest option is to have a second version of your model which is 'pre-rotated' by 90' on the x axis. That will allow your z rotation to function as you expect. That might not be the solution you need in this case, but it is a common solution to the problem.

Another option is to use the alternate skin for the Fixture in the stage patch and then connect to the x,y,z,w Quat ports. I believe these allow you to rotate the fixture using quaternion rotation. Only problem is I just tried this and I have no idea what data you need to feed into it. I know this solves this problem though if you can work out what to feed into it. Ciaran may be able to

elaborate here as he added these ports. There is a short video explaining quaternion rotation here: <https://www.youtube.com/watch?v=SCbpxiCN0U0> you should be able to take some math / trig modules in salvation and replicate the conversion detailed in this video if you are feeling brave ;)

And while I was still struggling to find my way through all this Ciaran was so kind to create a patch which is now included as system patch:

As Dave has said you encounter this problem based on the way Euler angles are defined in Euclidean Space. To get around this we can either use a 4x4 rotation matrix or quaternions. Without getting into the maths too much I have created a patch that should convert your Euler Angles into Quaternions, giving you back full 3 degrees of rotation.



From:

<https://avosupport.de/wiki/> - **AVOSUPPORT**

Permanent link:

<https://avosupport.de/wiki/ai/patches/eulertoquaternion>

Last update: **2018/12/04 16:20**

