

3D Rotation

Note: This section is currently under revision.

3D Rotation can be accomplished in a number of ways. Computer systems often favor Quaternions for certain mathematical properties. However Quaternions are not terribly easy for humans to interpret or understand specific values.

It must be stated that rotational systems are all mathematically consistent and equally valid. The method discussed here is simply easier for use of humans.

3x3 Rotation

This system describes an arbitrary rotation in 3D space with roll, pitch, and yaw, labeled ψ , ϕ , and θ .

Yaw θ describes rotation about z-axis. Pitch ϕ describes rotation about the y-axis. Roll ψ describes rotation about the x-axis. All rotations described here are right-handed.

These three values can be used to generate a 3×3 orthonormal matrix, with a determinant of 1, that rotates any $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ vector.

Roll

To Roll a vector about the x-axis, left-multiply it by the rotation vector R_ψ .

$$R_\psi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) \\ 0 & \sin(\psi) & \cos(\psi) \end{bmatrix}$$

Pitch

To Pitch a vector about the y-axis, left-multiply it by the rotation vector R_ϕ .

$$R_\phi = \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix}$$

Yaw

To Pitch a vector about the y-axis, left-multiply it by the rotation vector R_θ .

$$R_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Complete Rotation

A full 3D rotation includes a roll, pitch, and yaw. With these three rotations, we can describe any arbitrary orientation.

Order of operation is important. The complete R matrix describes the vehicle first yawing around its own z-axis, then pitching along its own y-axis, and then finally rolling about its own x-axis. As an example, the rotation $R([180,10,30])$ would have the submarine pointed to the left 30° , then pitched slightly upwards by 10° , and then rolling onto its back at 180° .

However, our rotation matrices do not provide rotations about our vehicle's intrinsic axes. They rotate vectors about the global, static x,y,z axes. Thus, to achieve a complete rotation, the vector must be first rolled, then pitched, then yawed, relative to these constant axes. These matrices are meant to operate on 3×1 column vectors on right right-hand side. As such, order of operation goes from right to left as more rotations are tacked onto the system.

$$R_{\{\psi,\phi,\theta\}} = R_{\theta} R_{\phi} R_{\psi} \quad R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) \\ 0 & \sin(\psi) & \cos(\psi) \end{bmatrix} \quad R([\psi,\phi,\theta]) = \begin{bmatrix} \cos(\theta)\sin(\phi) & \cos(\theta)\sin(\phi)\sin(\psi) - \sin(\theta)\cos(\psi) & \cos(\theta)\sin(\phi)\cos(\psi) + \sin(\theta)\sin(\psi) \\ \sin(\theta)\cos(\phi) & \sin(\theta)\sin(\phi)\sin(\psi) + \cos(\theta)\cos(\psi) & \sin(\theta)\sin(\phi)\cos(\psi) - \cos(\theta)\sin(\psi) \\ -\sin(\phi) & \cos(\phi)\sin(\psi) & \cos(\phi)\cos(\psi) \end{bmatrix}$$

Additionally, it is important to be able to reverse the process, and identify which combination of roll, pitch, and yaw describes the current orientation. This will be a non-unique combination of values, as there are an arbitrary number of ways to reach a given orientation. However, this process will yield consistent results.

$$\theta = \arctan(R_{21}, R_{11}), \quad \phi = \arctan(-R_{31}, \sqrt{R_{32}^2 + R_{33}^2}), \quad \psi = \arctan(R_{32}, R_{33})$$

In the event that $\phi = \pm 90^\circ$ the other values must be determined using the more complicated four elements in the upper-right corner. When pitched by $\pm 90^\circ$ yaw and roll become meaningless as independent values - only their sum or difference remain. Thus we can describe the orientation vector as $\begin{bmatrix} 0 & \phi & \theta^* \end{bmatrix}$ or as $\begin{bmatrix} \psi^* & \phi & 0 \end{bmatrix}$. We will go with the first version, reporting the roll parameter as equal to zero. For an arbitrary rotation including a pitch of $\phi = +90^\circ$ the yaw is reported as $\theta^* = \theta - \psi$. For an arbitrary rotation including a pitch of $\phi = -90^\circ$ the yaw is reported as $\theta^* = \theta + \psi$.

Looking at the formulation for the elements of R above and assuming $\phi = \pm 90^\circ$ we can find how to reverse-calculate our θ^*

$$\theta^* = \arctan(-R_{12}, R_{22})$$

Inverse Rotation

To reverse the rotation of an $\begin{bmatrix} x,y,z \end{bmatrix}$ matrix, you multiply it by the

$$= R_1^{\mathrm{T}} R_2 \quad \text{\$}$$

To find the rotation *between* two rotations, it is helpful to ask the question “What rotation would I need to achieve R_2 if R_1 was at the origin?” The answer is, of course, just R_2 . We find our relative motion by first un-rotating R_2 by R_1

Special Properties

Rotation matrices have some very exploitable properties. It stands to reason that there is an underlying structure, as we are using 9 elements to represent only 3 unique values.

Othanormal

A Rotation matrix's Transpose is equal to its inverse.

$$1 = \det(I) = \det(R^{\mathrm{T}} R) = \det(R^{\mathrm{T}}) \det(R) = (\det(R))^2 \quad \text{\$}$$

Code

To produce a 3×3 rotation matrix from roll ψ , pitch ϕ , and yaw θ use the following matlab code or it's C++ equivalent:

```
function [R] = r3D(omega) % omega = [roll; pitch; yaw]

R = [cosd(omega(3)) -sind(omega(3)) 0;...
     sind(omega(3)) cosd(omega(3)) 0;...
     0 0 1] * ...
     [cosd(omega(2)) 0 sind(omega(2));...
     0 1 0;...
     -sind(omega(2)) 0 cosd(omega(2))] * ...
     [1 0 0;...
     0 cosd(omega(1)) -sind(omega(1));...
     0 sind(omega(1)) cosd(omega(1))];

end
```

To find an equivalent roll ψ , pitch ϕ , and yaw θ given a 3×3 rotation matrix R use the following code:

```
function [omega] = ir3D(R)

theta = atan2di(R(2,1),R(1,1));
psi = atan2di(R(3,2),R(3,3));
phi = atan2di(-R(3,1),sqrt(sum(R(3,2:3).^2)));
```

```
if(abs(phi)==90)
    theta = atan2di(-R(1,2),R(2,2));
    psi = 0;
end

omega = [psi;phi;theta]; %omega = [roll;pitch;yaw]
end
```

From:

<https://robosub-vm.eecs.wsu.edu/wiki/> - **Palouse RoboSub Technical Documentation**

Permanent link:

<https://robosub-vm.eecs.wsu.edu/wiki/cs/localization/rotation/start>



Last update: **2018/04/29 11:25**