Subject: Re: 3D vector rotation to the Z axis
Posted by penteado on Tue, 14 Dec 2010 03:23:01 GMT
View Forum Message <> Reply to Message

On Dec 14, 1:11 am, MartyL <mlandsf...@yahoo.com> wrote:
> I would like to rotate and arbitrary 3D vector to be aligned with the
> Z axis. I can translate and scale the vector but I can not find a
> formula to create the rotation matrix to perform the rotation. It
> sounds simple but I can't seem to find one.
> Thanks

Do you mean something like

```
IDL> t3d,identity(4),matrix=matrix,rotate=[45d0,0d0,0d0]
IDL> print,matrix
      1.0000000      0.0000000      0.0000000      0.0000000
      0.0000000     0.70710678    -0.70710678      0.0000000
      0.0000000     0.70710678     0.70710678      0.0000000
      0.0000000      0.0000000      0.0000000      1.0000000
```

Subject: Re: 3D vector rotation to the Z axis
Posted by MartyL on Tue, 14 Dec 2010 20:20:48 GMT
View Forum Message <> Reply to Message

On Dec 13, 7:23 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On Dec 14, 1:11 am, MartyL <mlandsf...@yahoo.com> wrote:
>
>> I would like to rotate and arbitrary 3D vector to be aligned with the
>> Z axis. I can translate and scale the vector but I can not find a
>> formula to create the rotation matrix to perform the rotation. It
>> sounds simple but I can't seem to find one.
>> Thanks
>
> Do you mean something like
>
> IDL> t3d,identity(4),matrix=matrix,rotate=[45d0,0d0,0d0]
> IDL> print,matrix
>   1.0000000   0.0000000   0.0000000   0.0000000
>   0.0000000   0.70710678   -0.70710678   0.0000000
>   0.0000000   0.70710678   0.70710678   0.0000000
>   0.0000000   0.0000000   0.0000000   1.0000000

Yes, I am trying to get the transformation matrix for an arbitrary 3D vector. So the problem is that I need to find the degrees which to rotate. If I have the vector:

pts = [[5.1,20.4,-12.2],[15.0,55.5,30.4]]

I can translate that to the origin

pts_trans = [[0.0, 0.0, 0.0],[9.9,35.1,42.6]]

and normalize it to a unit vector

pts_norm = [[0.0, 0.0, 0.0],[0.176,0.625,0.759]]

but then I need to determine the rotation angles to rotate it to [0.0, 0.0, 1.0] or the Z axis.

I have found this solution in another news group but it did not produce the correct results.

Let V1 = [x1, y1, z1], V2 = [x2, y2, z2]. Assume V1 and V2 are already normalized.

V3 = normalize(cross(V1, V2)). (the normalization here is mandatory.)
V4 = cross(V3, V1).

```
    [ V1 ]
M1 = [ V4 ]
    [ V3 ]
```

cos = dot(V2, V1), sin = dot(V2, V4)

```
    [ cos   sin    0 ]
M2 = [ -sin  cos    0 ]
    [ 0     0      1 ]
```

The sought transformation matrix is just M1^-1 * M2 * M1. This might well be a standard-text solution.

I used [0.176,0.625,0.759], from pts_norm, as V1 and [0.0, 0.0, 1.0], the Z axis, as V2.
My implementation is this:

```
v1 = [0.176,0.625,0.759D]
v2 = [0.0, 0.0, 1.0D]

v = CROSSP(v1, v2)
v3 = v/NORM(v)
v4 = CROSSP(v3, v1)
```

```
m1 = DBLARR(3,3)
m1[*,0] = [v1]
m1[*,1] = [v4]
m1[*,2] = [v3]

rot_cos = TRANSPOSE(v1)#v2
rot_sin = TRANSPOSE(v2)#v4

m2 = DBLARR(3,3)
m2[*,0] = [rot_cos, rot_sin, 0.0]
m2[*,1] = [-rot_sin, rot_cos, 0.0]
m2[*,2] = [0.0, 0.0, 1.0]

m = MATRIX_MULTIPLY(INVERT(m1), m2)
m4 = MATRIX_MULTIPLY(m, m1)

v1_trans = MATRIX_MULTIPLY(v1, m4)
print, v1_trans
```

IDL output is:

```
 -0.0875152
  0.933033
  0.341724
```

which should have been:

```
  0.0
  0.0
  1.0
```

I am wondering if I have the order of vectors and matrices wrong in MATRIX_MULTIPLY.
Any help appreciated.

Subject: Re: 3D vector rotation to the Z axis
Posted by David Fanning on Tue, 14 Dec 2010 20:28:14 GMT

View Forum Message <> Reply to Message

MartyL writes:

> Any help appreciated.

I don't have any suggestions. You obviously know more about this than I do. Just an observation. I know any time I try to translate matrix operations out of a book into IDL my head swells up to about three times its

normal size trying to keep columns and rows straight.

That said, using the # operator, rather than the ## operator always throws me back from discovering the solution at LEAST two days, sometimes more. :-(

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: 3D vector rotation to the Z axis
Posted by James[2] on Thu, 16 Dec 2010 19:48:18 GMT
View Forum Message <> Reply to Message

Do you have to use a rotation matrix? Quaternions (http://en.wikipedia.org/wiki/Quaternion) are more numerically stable, and there is an easy-to-use library by Craig Markwardt at http://www.physics.wisc.edu/~craigm/idl/math.html. I was just using them to draw some rotating polyhedra in IDL, and they work great! Your code would look like this:

```
;define the axis of rotation
rotaxis = crossp (input, [0,0,1])

;find the angle of rotation
rotangle = transpose(input) # [0,0,1]

;make the quaternion
q = qtcompose(rotaxis, rotangle)

;do it
rotated = qtvrot(input, q)

;there is even a routine to create a rotation matrix from the
quaternion:
rmatx = qtmat(q)
```

by the way, I like the # operator. It lets you treat 1D arrays as column vectors; then defining a matrix is just concatenating a group of column vectors across the second dimension. Matrix-by-vector

multiplication works like you expect.

- James

On Dec 14, 12:28 pm, David Fanning <n...@dfanning.com> wrote:
> MartyL writes:
>> Any help appreciated.
>
> I don't have any suggestions. You obviously know more
> about this than I do. Just an observation. I know any
> time I try to translate matrix operations out of a book
> into IDL my head swells up to about three times its
> normal size trying to keep columns and rows straight.
>
> That said, using the # operator, rather than the ## operator
> always throws me back from discovering the solution at LEAST
> two days, sometimes more. :-(
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: 3D vector rotation to the Z axis
Posted by MartyL on Tue, 04 Jan 2011 01:17:49 GMT
View Forum Message <> Reply to Message

On Dec 16 2010, 11:48 am, James <donje...@gmail.com> wrote:
> Do you have to use a rotation matrix? Quaternions (http://
> en.wikipedia.org/wiki/Quaternion) are more numerically stable, and
> there is an easy-to-use library by Craig Markwardt
athttp://www.physics.wisc.edu/~craigm/idl/math.html. I was just using
> them to draw some rotating polyhedra in IDL, and they work great!
> Your code would look like this:
>
> ;define the axis of rotation
> rotaxis = crossp (input, [0,0,1])
>
> ;find the angle of rotation
> rotangle = transpose(input) # [0,0,1]
>

> ;make the quaternion
> q = qtcompose(rotaxis, rotangle)
>
> ;do it
> rotated = qtvrot(input, q)
>
> ;there is even a routine to create a rotation matrix from the
> quaternion:
> rmatx = qtmat(q)
>
> by the way, I like the # operator. It lets you treat 1D arrays as
> column vectors; then defining a matrix is just concatenating a group
> of column vectors across the second dimension. Matrix-by-vector
> multiplication works like you expect.
>
> - James
>

James,
I do not need to use a rotation matrix per se, I just need to rotate
the vectors to the Z axis.
I copied your code and made a quick test procedure and I am not
getting the results I would expect. Here is the code I used:

```
PRO Rotate_to_Z_axis, input

  print, 'input = ', input
  ;define the axis of rotation
  rotaxis = crossp (input, [0,0,1])

  ;find the angle of rotation
  rotangle = transpose(input) # [0,0,1]

  ;make the quaternion
  q = qtcompose(rotaxis, rotangle)

  ;do it
  rotated = qtvrot(input, q)
  print, 'rotated = ', rotated

end
```

I then input the following at the command prompt:

```
rotate_to_z_axis, [cos(!dtor*30.0), sin(!dtor*30.0), 0.0]
```

and the results are:

input = 0.866025 0.500000 0.00000
rotated = 0.866025 0.500000 0.00000

Shouldn't the rotated vector be [0,0,1] if it is being rotated to the
Z axis? That is what I am trying to do at least.
