Robótica taller 1

Daniel Esteban Ramirez Chiquillo

dramirezch@unal.edu.co

c.c. 1002479235

1

Tenemos dos vectores libres A y B expresados en un marco de referencia M1.

A = [a1, a2, a3]

B = [b1, b2, b3]

La representación de estos vectores en un marco de referencia M2 está dada por:

A' = Tm A = [a1', a2', a3']

B' = Tm B = [b1', b2', b3']

En donde Tm es una matriz de transformación.

Calculamos el producto punto de A' y B':

A' dot B' = (Tm A)^T (Tm B)

A' dot B' = Tm^T A^T Tm B

Como Tm^T Tm = I para cualquier matriz de transformación Tm:

A' dot B' = A^T B

Recordamos la definición del producto punto de A y B:

A dot B = A^T B

Con esto queda demostrado que:

A dot B = A' dot B'

Por lo que el producto punto de dos vectores libres no depende del los marcos de referencia seleccionados.

2

¿Qué es una base?

Una base es un conjunto de vectores linealmente independientes que pueden representar todos los vectores de un espacio vectorial específico mediante una combinación lineal.

¿Qué es un sistema de coordenadas?

Un sistema de coordenadas es un sistema que utiliza uno o más números, o coordenadas, para determinar la posición única de los puntos u otros elementos geométricos en un espacio euclidiano.

3

a)

```
w = [0, -4]';
S = [[1, -3]', [2, -2]'];
x = S \ w
```

```
x = 2×1
    2.0000
   -1.0000
```

b)

```
W = [6 -5; 1 -2];
S = [[1 -1; 1 0],[-1 1; 0 1],[1 0; -1 1]];
X = S \ W
```

```
X = 6×2
    3.5000   -3.5000
         0         0
         0         0
         0         0
    2.5000   -1.5000
         0         0
```

4

```
i = [1 0 0]';
j = [0 1 0]';
k = [0 0 1]';
base_canonica = [i,j,k]
```

```
base_canonica = 3×3
     1     0     0
     0     1     0
     0     0     1
```

Para obtener la combinación lineal se realiza el mismo procedimiento que en el ejercicio anterior.

```
d = [4 2 -1]';
clvbc = base_canonica \ d
```

```
clvbc = 3×1
     4
     2
    -1
```

Las proyeciones del vector en la base son los productos punto del vector con los vectores de la base. (Multiplicado por el vector de la base para darle dirección.)

```
proy_i = dot(d, i) * i
```

```
proy_i = 3×1
     4
     0
     0
```

```
proy_j = dot(d, j) * j
```

```
proy_j = 3×1
     0
     2
     0
```

```
proy_k = dot(d, k) * k
```

```
proy_k = 3×1
     0
     0
    -1
```

Teniendo en cuenta la formula:

A dot B = |A| |B| cos(theta)

cos(theta) = |A| |B| / (A dot B)

```
cos_i = norm(proy_i) / (norm(i)*norm(d))
```

```
cos_i = 0.8729
```

```
cos_j = norm(proy_j) / (norm(j)*norm(d))
```

```
cos_j = 0.4364
```

```
cos_k = norm(proy_k) / (norm(k)*norm(d))
```

```
cos_k = 0.2182
```

Gráfica:

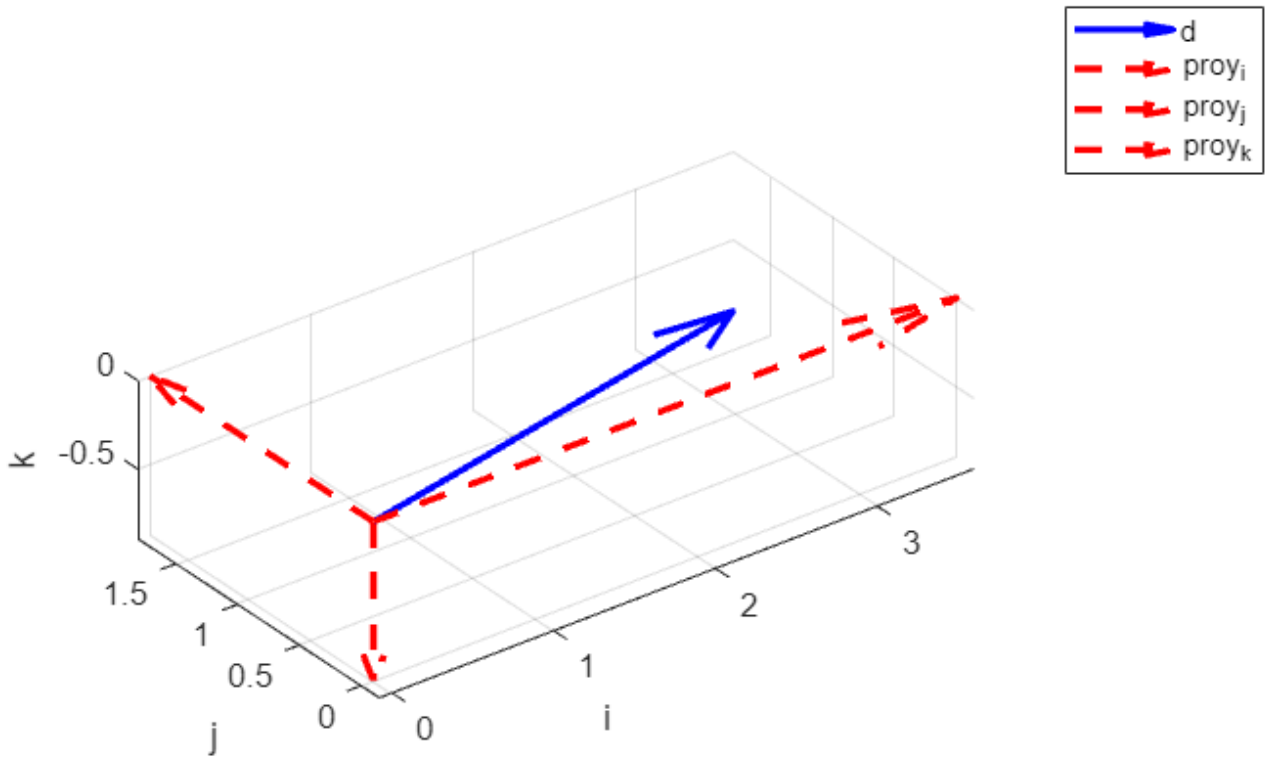
```
% quiver3 plotea un vector 3D de un punto x,y,z a otro x2,y2,z2
% se grafica el vector d
quiver3(0, 0, 0, d(1), d(2), d(3), 'b', 'LineWidth', 2, 'MaxHeadSize', 0.5);
hold on;
% se grafican las proyecciones
quiver3(0, 0, 0, proy_i(1), proy_i(2), proy_i(3), 'r--', 'LineWidth', 2, 'MaxHeadSize', 0.5);
quiver3(0, 0, 0, proy_j(1), proy_j(2), proy_j(3), 'r--', 'LineWidth', 2, 'MaxHeadSize', 0.5);
quiver3(0, 0, 0, proy_k(1), proy_k(2), proy_k(3), 'r--', 'LineWidth', 2, 'MaxHeadSize', 0.5);

% límites y tamaños de los ejes
xlim([-5, 5]);
ylim([-5, 5]);
zlim([-5, 5]);
axis equal;

% labels para los ejes
xlabel('i');
ylabel('j');
zlabel('k');

% leyenda
```

```
legend('d', 'proy_i', 'proy_j', 'proy_k');
hold off
```



5

```
A = [-5 2 10; -3 1 -6; 1 -3 3];
B = [-2 2; 3 -1; -3 2];
C = [1 0 -1; 0 1 1];
D = [-15 24 -2; 3 -5 0; 8 -13 1];
```

(Se asume que se pide realizar multiplicación de matrices y no producto punto como dice el enunciado.)

a)

```
ans_5a = (A * B)'
```

```
ans_5a = 2×3
   -14    27   -20
     8   -19    11
```

b)

```
ans_5b = D' * A'
```

```
ans_5b = 3×3
   161     0     0
  -260     1     0
    20     0     1
```

c)

No se puede realizar: (B' * B)^-1) da como resultado una matriz 2x2 y esta no se puede multiplicar con una matriz 3x2.

```
bb = (B' * B)^-1
```

```
bb = 2×2
    0.3103    0.4483
    0.4483    0.7586
```

```
bt = B'
```

```
bt = 2×3
    -2     3    -3
     2    -1     2
```

d)

No se puede realizar: B*C es una matriz 3x3 y C*B es 2x2. Estas dos no se pueden restar

```
bc = (B * C)
```

```
bc = 3×3
    -2     2     4
     3    -1    -4
    -3     2     5
```

```
cb = (C * B)
```

```
cb = 2×2
     1     0
     0     1
```

e)

(Es la misma que la anterior.)

f)

```
ans_5f = (B' * (2*B)) * (C * B)
```

```
ans_5f = 2×2
    44   -26
   -26    18
```

6

a)

```
R1 = [0.4330 -0.75 -0.5; 0.4356 0.6597 -0.6124; 0.7891 0.0474 0.6124];
```

Ortogonal

Si la matriz es ortogonal, la multiplicación de ella misma con su transpuesta debe dar como resultado la matriz identidad.

```
R1*R1'
```

```
ans = 3×3
    1.0000    0.0000   -0.0001
    0.0000    1.0000   -0.0000
   -0.0001   -0.0000    1.0000
```

Por lo tanto, la matriz es **ortogonal**. (Los decimales que aparecen en la diagonal secundaria son errores de aproximación de Matlab.)

Determinante igual a 1

```
det(R1)
```

```
ans = 1.0000
```

Esto comprueba otra propiedad, que su **determinante es igual a 1**.

Transpuesta = inversa

```
R1'
```

```
ans = 3×3
    0.4330    0.4356    0.7891
   -0.7500    0.6597    0.0474
   -0.5000   -0.6124    0.6124
```

```
R1^-1
```

```
ans = 3×3
    0.4330    0.4356    0.7892
   -0.7500    0.6597    0.0474
   -0.4999   -0.6124    0.6124
```

Se puede ver que las dos matrices son iguales, por lo tanto, l**a transpuesta y la inversa de la matriz son iguales**.

b)

Las imprecisiones en los cálculos pueden hacer que el determinante se calcule como 0 cuando no debería ser así, llevando a la conclusión de que la matriz es singular y no tiene inversa. Del mismo modo, las imprecisiones en el cálculo de la matriz inversa pueden conducir a resultados incorrectos al resolver sistemas lineales de ecuaciones.

El comando que genera la cantidad más pequeña con la que Matlab puede trabajar es:

```
eps
```

```
ans = 2.2204e-16
```

7

```
R_B_A = [1 0 0; 0 (sqrt(3)/2) -1/2; 0 1/2 (sqrt(3)/2)];
R_C_A = [0 0 -1; 0 1 0; 1 0 0];
```

Para obtener la tercera matriz de rotación se hace la siguiente operación:

```
R_C_B = R_B_A' * R_C_A
```

```
R_C_B = 3×3
         0         0   -1.0000
    0.5000    0.8660         0
    0.8660   -0.5000         0
```

8

```
R_B_A = [1 0 0; 0 (sqrt(3)/2) -1/2; 0 1/2 (sqrt(3)/2)]
```

```
R_B_A = 3×3
    1.0000         0         0
         0    0.8660   -0.5000
         0    0.5000    0.8660
```

Esta es una matriz de rotación respecto al eje x:

```
pitch = 0
```

```
pitch = 0
```

```
yaw = 0
```

```
yaw = 0
```

Roll será el ángulo de giro dado por:

```
roll = rad2deg(atan(R_B_A(3,2)/R_B_A(3,3)))
```

```
roll = 30.0000
```

9

```
R_B_A = [0.4330 -0.25 0.866; 0.8839 0.3062 -0.3536; -0.1768 0.9186 0.3536]
```

```
R_B_A = 3×3
```

```
    0.4330   -0.2500    0.8660
    0.8839    0.3062   -0.3536
   -0.1768    0.9186    0.3536
```

```
psi = rad2deg(atan2(R_B_A(2,1), R_B_A(1,1)))
```

```
psi = 63.9009
```

```
theta = rad2deg(atan2(-R_B_A(3,1), sqrt(R_B_A(1,1)^2 + R_B_A(2,1)^2)))
```

```
theta = 10.1833
```

```
phi = rad2deg(atan2(R_B_A(3,2), R_B_A(3,3)))
```

```
phi = 68.9466
```

10

```
k=1/sqrt(3)*[1 1 1]';
theta=90;

v_theta = 1 - cosd(theta)
```

```
v_theta = 1
```

```
R_K =[(k(1)*k(1)*v_theta + cosd(theta)), (k(1)*k(2)*v_theta - k(3)*sind(theta)), (k(1)*k(3)*v_t
```

```
R_K = 3×3
    0.3333   -0.2440    0.9107
    0.9107    0.3333   -0.2440
   -0.2440    0.8495    0.3333
```

11

```
theta_1 = 45;
primera_rotacion = [cosd(theta_1) 0 sind(theta_1); 0 1 0; -sind(theta_1) 0 cosd(theta_1)]
```

```
primera_rotacion = 3×3
    0.7071         0    0.7071
         0    1.0000         0
   -0.7071         0    0.7071
```

```
theta_2 = 30;
segunda_rotacion = [cosd(theta_2) -sind(theta_2) 0; sind(theta_2) cosd(theta_2) 0; 0 0 1]
```

```
segunda_rotacion = 3×3
    0.8660   -0.5000         0
    0.5000    0.8660         0
         0         0    1.0000
```

```
R = primera_rotacion * segunda_rotacion
```

```
R = 3×3
    0.6124   -0.3536    0.7071
    0.5000    0.8660         0
   -0.6124    0.3536    0.7071
```

Toolbox

```
R_tb = troty(deg2rad(45))*trotz(deg2rad(30))
```

```
R_tb = 4×4
    0.6124   -0.3536    0.7071         0
    0.5000    0.8660         0         0
   -0.6124    0.3536    0.7071         0
         0         0         0    1.0000
```

```
rpy = tr2rpy(R_tb)
```

```
rpy = 1×3
    0.4636    0.6591    0.6847
```

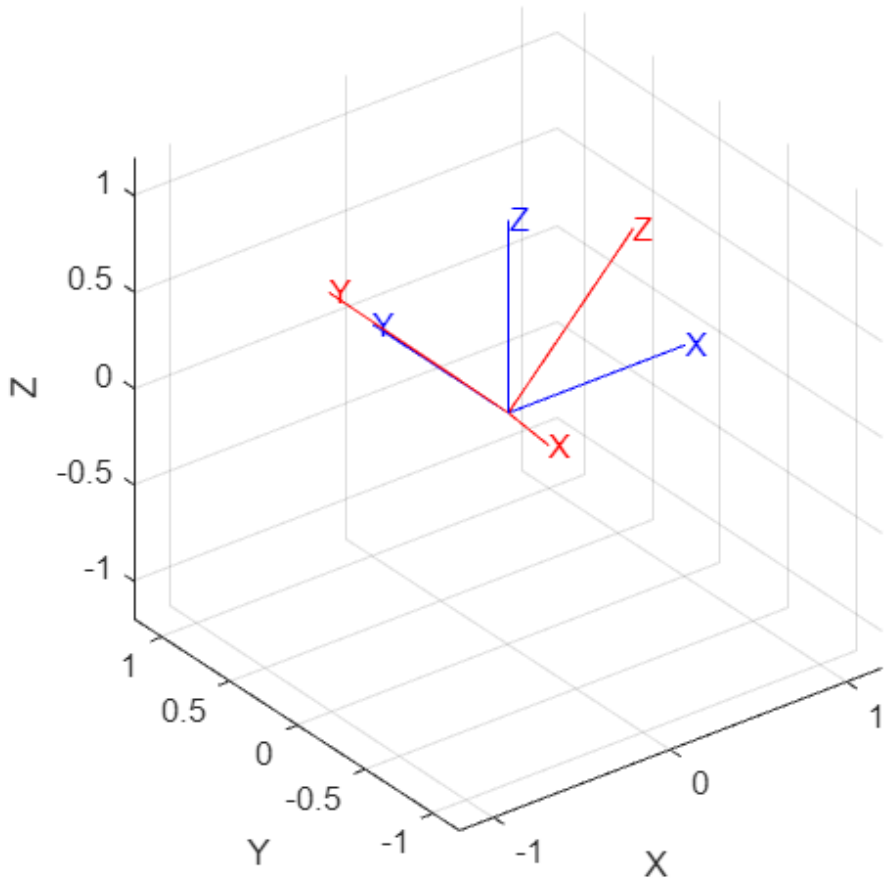
```
eul = tr2eul(R_tb)
```

```
eul = 1×3
         0    0.7854    0.5236
```

```
[theta, vec] = tr2angvec(R_tb)
```

```
theta = 0.9363
vec = 1×3
    0.2195    0.8192    0.5299
```

```
trplot(eye(4))
hold on
trplot(R_tb, 'color', 'r')
hold off
```



12

```
T10 = mth_trans_x(1) * mth_trans_y(0.5) * mth_rot_y(180)
```

```
T10 = 4×4
   -1.0000         0         0    1.0000
         0    1.0000         0    0.5000
         0         0   -1.0000         0
         0         0         0    1.0000
```

```
T20 = mth_trans_z(1) * mth_rot_x(-90) * mth_rot_z(90)
```

```
T20 = 4×4
     0    -1     0     0
     0     0     1     0
    -1     0     0     1
     0     0     0     1
```

```
T21 = mth_trans_x(1) * mth_trans_y(-0.5) * mth_trans_z(-1) * mth_rot_x(-90) * mth_rot_z(-90)
```

```
T21 = 4×4
         0    1.0000         0    1.0000
```

```
         0         0    1.0000   -0.5000
    1.0000         0         0   -1.0000
         0         0         0    1.0000
```

```
T20_dem = T10 * T21
```

```
T20_dem = 4×4
     0    -1     0     0
     0     0     1     0
    -1     0     0     1
     0     0     0     1
```

Toolbox

```
T10tb = rt2tr(rotx(0, "deg")*roty(180, "deg")*rotz(0, "deg"), [1 0.5 0])
```

```
T10tb = 4×4
   -1.0000         0         0    1.0000
         0    1.0000         0    0.5000
         0         0   -1.0000         0
         0         0         0    1.0000
```

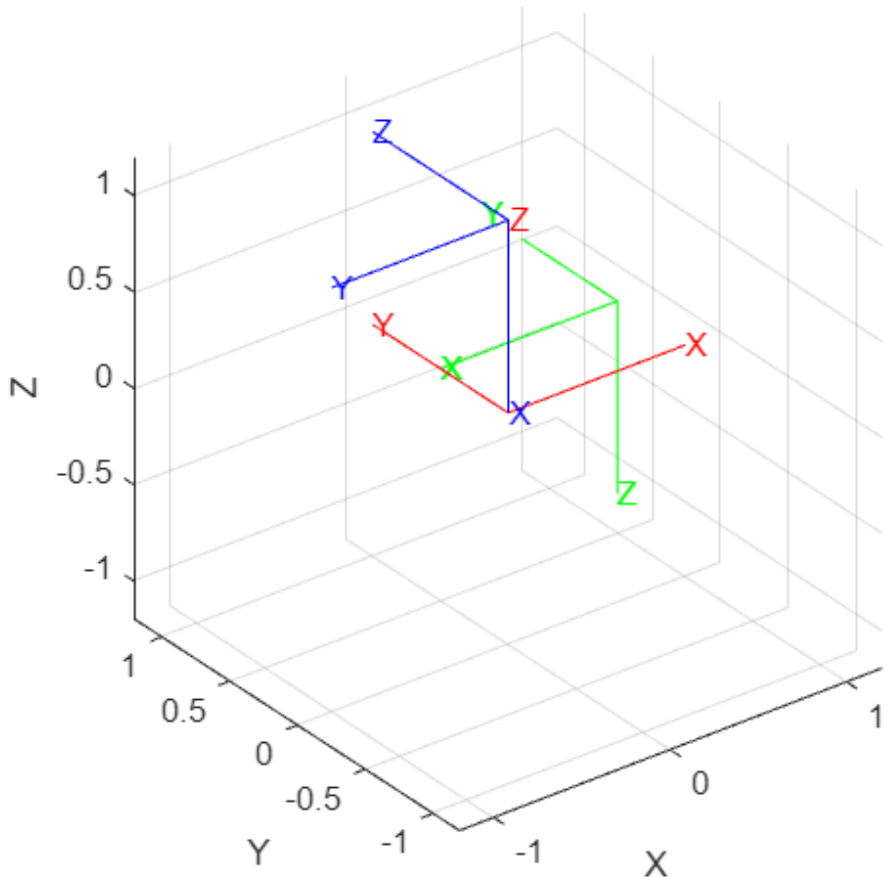
```
T20tb = rt2tr(rotx(-90, "deg")*roty(0, "deg")*rotz(90, "deg"), [0 0 1])
```

```
T20tb = 4×4
     0    -1     0     0
     0     0     1     0
    -1     0     0     1
     0     0     0     1
```

```
T21tb = rt2tr(rotx(-90, "deg")*roty(0, "deg")*rotz(-90, "deg"), [1 -0.5 -1])
```

```
T21tb = 4×4
         0    1.0000         0    1.0000
         0         0    1.0000   -0.5000
    1.0000         0         0   -1.0000
         0         0         0    1.0000
```

```
trplot(eye(4),'color','r')
hold on;
trplot(T10,'color','g')
trplot(T20,'color','b')
hold off
```



13

```
Tda = mth_trans_x(-1) * mth_trans_y(1)
```

```
Tda = 4×4
     1     0     0    -1
     0     1     0     1
     0     0     1     0
     0     0     0     1
```

```
Tdc = mth_trans_z(2) * mth_rot_x(180) * mth_rot_z(90)
```

```
Tdc = 4×4
     0    -1     0     0
    -1     0     0     0
     0     0    -1     2
     0     0     0     1
```

```
Tcb = [1 0 0 -2; 0 1 0 1; 0 0 1 0; 0 0 0 1];

Tba = Tda* mth_inv(Tdc) * mth_inv(Tcb)
```

```
Tba = 4×4
     0    -1     0     0
    -1     0     0    -1
     0     0    -1     2
     0     0     0     1
```

```
Tca = mth(-1,1,2, 180,0,-90);

tr2eul(Tba)
```

```
ans = 1×3
         0    3.1416   -1.5708
```

```
tr2rpy(Tba)
```

```
ans = 1×3
    3.1416         0   -1.5708
```

```
tr2eul(Tca)
```

```
ans = 1×3
         0    3.1416    1.5708
```

```
tr2rpy(Tca)
```

```
ans = 1×3
    3.1416         0    1.5708
```

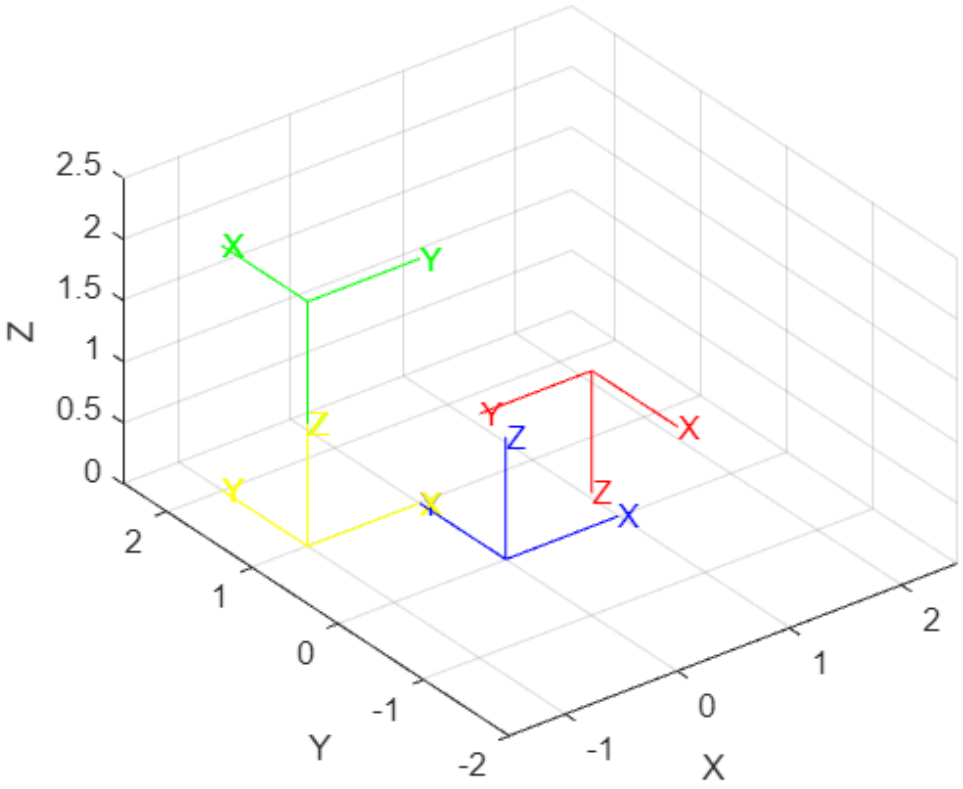
```
tr2eul(Tda)
```

```
ans = 1×3
     0     0     0
```

```
tr2rpy(Tda)
```

```
ans = 1×3
     0     0     0
```

```
trplot(eye(4), 'axis', [-1.5 2.5 -2 2.5 0 2.5])
hold on
trplot(Tba, 'color', 'r')
trplot(Tca, 'color', 'g')
trplot(Tda, 'color', 'y')
hold off
```



14

```
Tca = rt2tr(rotx(180, "deg")*roty(0, "deg")*rotz(53.1, "deg"), [-3 4 2])
```

```
Tca = 4×4
    0.6004   -0.7997         0   -3.0000
   -0.7997   -0.6004         0    4.0000
         0         0   -1.0000    2.0000
         0         0         0    1.0000
```

```
Tcb = rt2tr(rotx(90, "deg")*roty(0, "deg")*rotz(180+53.1, "deg"), [3 0 0])
```

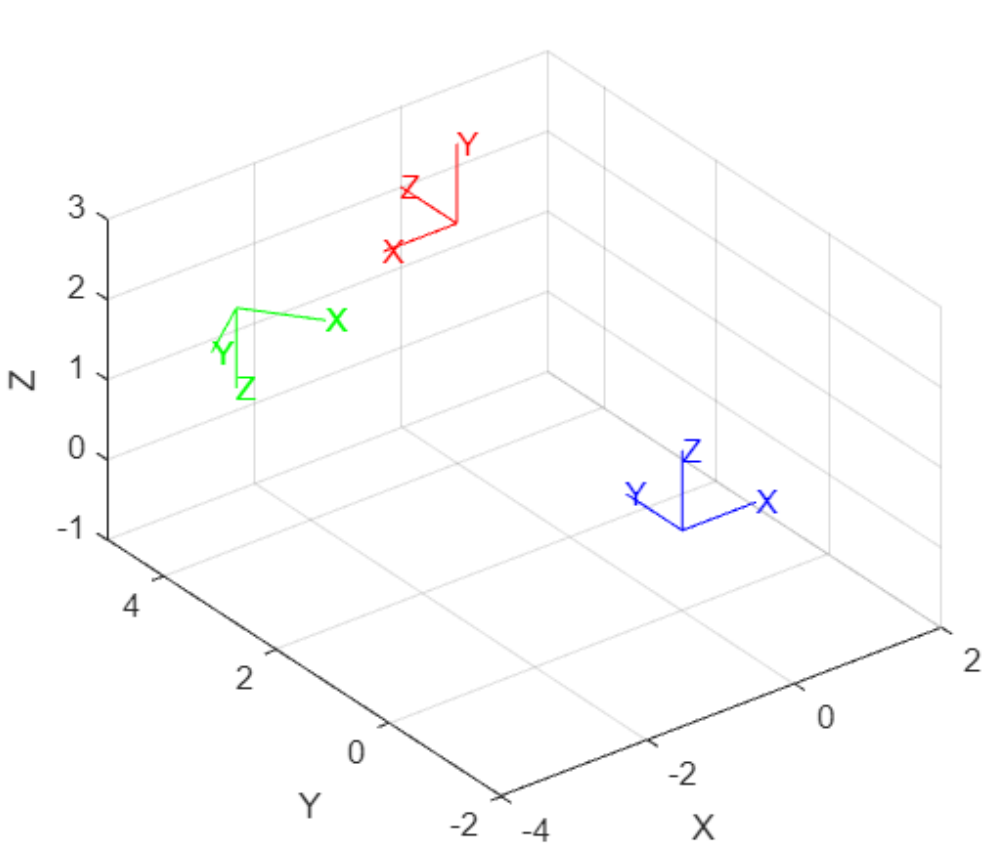
```
Tcb = 4×4
   -0.6004    0.7997         0    3.0000
         0         0   -1.0000         0
   -0.7997   -0.6004         0         0
         0         0         0    1.0000
```

```
Tac = rt2tr(rotx(180, "deg")*roty(0, "deg")*rotz(-36.9, "deg"), [3*cosd(36.9) 4*cosd(36.9) 2])
```

```
Tac = 4×4
    0.7997    0.6004         0    2.3991
    0.6004   -0.7997         0    3.1987
         0         0   -1.0000    2.0000
```

```
         0         0         0    1.0000
```

```
Tba = Tca * mth_inv(Tcb);
trplot(eye(4), 'axis', [-4 2 -2 5 -1 3])
hold on
trplot(Tba, 'color', 'r')
trplot(Tca, 'color', 'g')
hold off
```




```
tr2eul(Tba)
```

```
ans = 1×3
    1.5708    1.5708    1.5708
```

```
tr2rpy(Tba)
```

```
ans = 1×3
    1.5708         0    3.1416
```

```
tr2eul(Tca)
```

```
ans = 1×3
         0    3.1416   -2.2148
```

```
tr2rpy(Tca)
```

```
ans = 1×3
    3.1416         0   -0.9268
```

15

```
k = 1/sqrt(14) * [2, 3, 1]
```

```
k = 1×3
    0.5345    0.8018    0.2673
```

```
theta = pi/3;
v01 = 1-cos(theta);

R = [(k(1)*k(1)*v01 + cos(theta)), (k(1)*k(2)*v01 - k(3)*sin(theta)), (k(1)*k(3)*v01 + k(2)*sin
```

```
R = 3×3
    0.6429   -0.0172    0.7658
    0.4457    0.8214   -0.3558
   -0.6229    0.5701    0.5357
```

```
Rtb = angvec2r(theta,k)
```

```
Rtb = 3×3
    0.6429   -0.0172    0.7658
    0.4457    0.8214   -0.3558
   -0.6229    0.5701    0.5357
```

16

```
Q = [cos(theta/2); k(1)*sin(theta/2); k(2)*sin(theta/2); k(3)*sin(theta/2)]
```

```
Q = 4×1
    0.8660
    0.2673
    0.4009
    0.1336
```

```
Qtb = Quaternion(theta,k)
```

```
Qtb =

1.0472 << 0.53452, 0.80178, 0.26726 >>
```

Transformaciones Homogeneas Básicas

```
function [MTHtx] = mth_trans_x(dist)
    MTHtx = [1 0 0 dist; 0 1 0 0; 0 0 1 0; 0 0 0 1];
end

function [MTHty] = mth_trans_y(dist)
    MTHty = [1 0 0 0; 0 1 0 dist; 0 0 1 0; 0 0 0 1];
end
```

```
function [MTHtz] = mth_trans_z(dist)
    MTHtz = [1 0 0 0; 0 1 0 0; 0 0 1 dist; 0 0 0 1];
end

function [MTHrx] = mth_rot_x(ang)
    MTHrx = [1 0 0 0; 0 cosd(ang) -sind(ang) 0; 0 sind(ang) cosd(ang) 0; 0 0 0 1];
end

function [MTHry] = mth_rot_y(ang)
    MTHry = [cosd(ang) 0 sind(ang) 0; 0 1 0 0; -sind(ang) 0 cosd(ang) 0; 0 0 0 1];
end

function [MTHrz] = mth_rot_z(ang)
    MTHrz = [cosd(ang) -sind(ang) 0 0; sind(ang) cosd(ang) 0 0; 0 0 1 0; 0 0 0 1];
end

function [MTHt] = mth_trans(x,y,z)
    MTHt = mth_trans_x(x) * mth_trans_y(y) * mth_trans_z(z);
end

function [MTHr] = mth_rot(x,y,z)
    MTHr = mth_rot_x(x) * mth_rot_y(y) * mth_rot_z(z);
end

function [MTH] = mth(x,y,z, angx, angy, angz)
    MTH = mth_trans(x,y,z) * mth_rot(angx,angy,angz);
end

function [MTH_inv] = mth_inv(mth)
    R = mth(1:3, 1:3);
    d = mth(1:3, 4);
    MTH_inv = [R' -R'*d; 0 0 0 1];
end
```