Successive rotations demo

Table of Contents

Motivation

The purpose of this demo is to build intution behind the proces of constructing so called 'slave' coordinate system from 'master' coordinate by applying succesive rotations to the master and its derivatives. This demo shows explictly how the rotation matrix is build as the product of rotation matrices of successive rotations, and what is its relation to the matrix that allows to map coordinates from slave coordinate system to master coordinate system.

In this demo we will use four coordinate systems:

- csG -- global coordinate system
- csA -- master coordinate system
- csB -- intermediate coordinate system
- csC -- slave coordinate system

General script setup

```
clear variables;
```

Create coordinate systems

Global coordinate sysstem

```
csG = cs.CoordSys(struct('name', 'global', 'color', 'black'));
```

Master coordinate system is created from global one by rotation about 10 degees

```
csA = cs.make_rotated(10, csG);
csA.params.name = 'master';
csA.params.color = 'red';
```

Intermediate coordinate system is created from master by rotation about 20 degrees

```
csB = cs.make_rotated(20, csA);
csB.params.name = 'intermediate';
csB.params.color = 'green';
```

Finally slave coordinate system is created from intermediate by rotation about 30 degrees

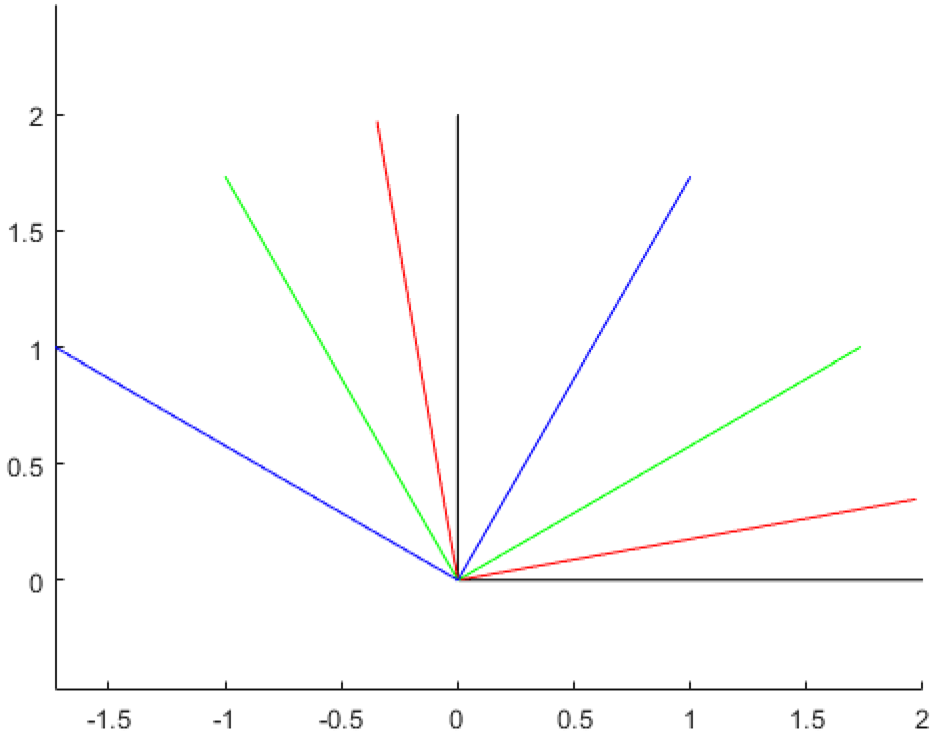
```
csC = cs.make_rotated(30, csB);
csC.params.name = 'slave';
csC.params.color = 'blue';
```

Create viewer and draw coordinate systems

The simplest way to visualize coordinate transformations is to use Viewer class.

```
viewer = cs.Viewer();

viewer.showCoordSys(csG);
viewer.showCoordSys(csA);
viewer.showCoordSys(csB);
viewer.showCoordSys(csC);
```



Illustrate coordinate transformation from slave to master

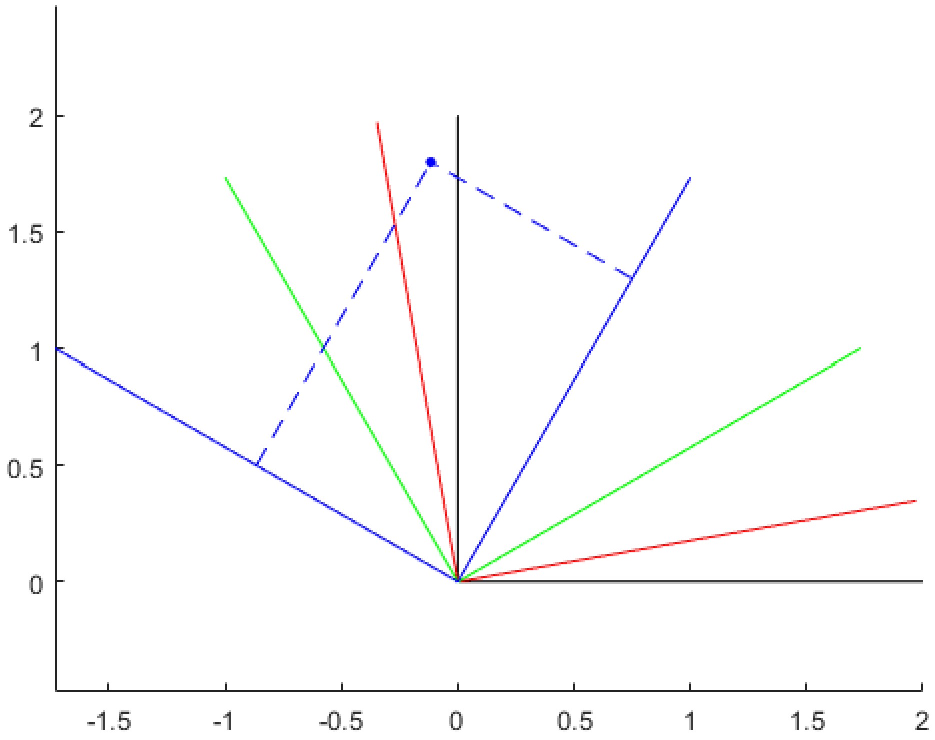
To illustrate coordinate transformation from slave to master we are going to pick a point in slave coordinate frame.

```
tpC = [1.5;1]
tpC =

    1.5000
    1.0000
```

We show it in slave coordsys (the blue one)

```
viewer.showPoint(tpC, csC, struct('filled', true, 'size', 15));
viewer.showPointProjections(tpC, csC);
```



Then we find matrix transforming from slvave (C) to master (A)

```
Q_c_a = cs.get_transformation_matrix(csC, csA)
```

```
Q_c_a =

    0.6428   -0.7660
    0.7660    0.6428
```

Using this matrix we can find coordinates of test point in coordinate system A

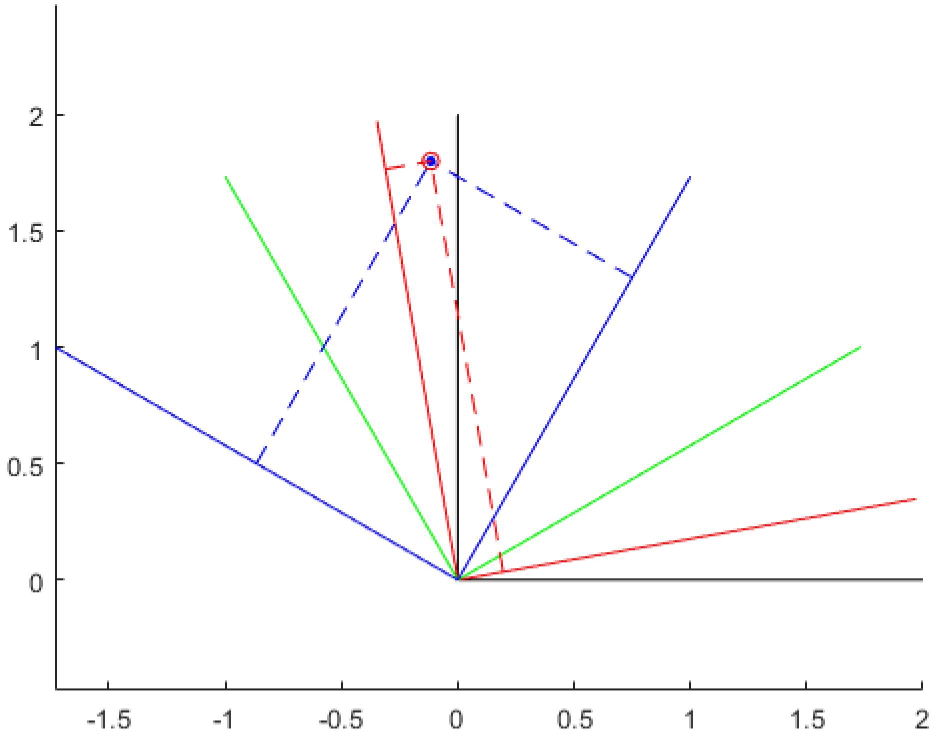
```
tpA = Q_c_a * tpC
tpA =

    0.1981
    1.7919
```

We can mark this point in coordinate system A

```
viewer.showPoint(tpA, csA, struct('filled', false, 'size', 40));
viewer.showPointProjections(tpA, csA);

cs_manage_demos('report', 'successive_rotations', true);
```



Published with MATLAB® R2020b