Aalto University
School of Electrical
Engineering

Time and motion; part 1

ELEC-C1320 Robotics

Pekka Forsman

Topics

- Time varying coordinate frames
  - Rotating coordinate frame
  - Incremental motion
- Inertial navigation systems
- *Example problems*

Time varying coordinate frames
Rotating coordinate frame

In chapter 2 of the text book we have learnt how to describe the pose of objects in 2- or 3-dimensional space. Chapter 3 extends those concepts to objects whose *pose is varying as a function of time.*

First we will discuss rotational velocity, especially in relation with a rotating coordinate frame

A body rotating in 3-dimensional space has an angular velocity which is a vector quantity $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)$. The direction of this vector defines the instantaneous axis of rotation, that is, the axis about which the coordinate frame is rotating at a particular instant of time. *In general this axis changes with time*. (Note: The magnitude of the vector is the rate of rotation about the axis – in this respect it is similar to the angle-axis representation for rotation)

The expression for the derivative of a time-varying rotation matrix, due to a angular velocity given with respect to frame {*A*}, is given as

$$^{A}\dot{\boldsymbol{R}}_{B} = \left[{}^{A}\boldsymbol{\omega}\right]_{\times} {}^{A}\boldsymbol{R}_{B} \in \mathbb{R}^{3\times 3} \qquad (3.1)$$

where $\left[{}^{A}\omega\right]_{X}$ is a *skew-symmetric matrix* (angular velocity matrix) that, for the 3-dimensional case, has the form

$$[\boldsymbol{\omega}]_{\times} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} \qquad (2.19)$$

In case the angular velocity is expressed w.r.t frame {*B*} we have

$$^{A}\dot{\boldsymbol{R}}_{B} = {}^{A}\boldsymbol{R}_{B} \left[{}^{B}\boldsymbol{\omega}\right]_{\times} \in \mathbb{R}^{3\times 3} \qquad (3.2)$$

If the pose (position and orientation) is given as a homogenous transformation matrix

$$\xi \sim {}^{A}\boldsymbol{T}_{B} = \begin{pmatrix} {}^{A}\boldsymbol{R}_{B} & {}^{A}\boldsymbol{t}_{B} \\ 0_{1\times 3} & 1 \end{pmatrix}$$

then, in case the angular velocity vector has been given w.r.t. frame {*A*}, we get for the derivative of the pose

$$\dot{\xi} \sim {}^{A}\dot{\boldsymbol{T}}_{B} = \begin{pmatrix} {}^{A}\dot{\boldsymbol{R}}_{B} & {}^{A}\dot{\boldsymbol{t}}_{B} \\ 0_{1\times 3} & 0 \end{pmatrix} = \begin{pmatrix} \left[{}^{A}\boldsymbol{\omega}\right]_{\times} {}^{A}\boldsymbol{R}_{B} & {}^{A}\dot{\boldsymbol{t}}_{B} \\ 0_{1\times 3} & 0 \end{pmatrix}$$

We can combine the two velocities into a velocity vector

$${}^{A}\nu_{B} = \left({}^{A}\boldsymbol{v}_{B}, {}^{A}\boldsymbol{\omega}_{B}\right) \in \mathbb{R}^{6}$$

which is the instantaneous velocity of frame {*B*} w.r.t. frame {*A*}

A first-order approximation to the derivative of the rotation matrix ***R*** can be written as

$$\dot{\boldsymbol{R}} \approx \frac{\boldsymbol{R}\langle t+\delta_t \rangle - \boldsymbol{R}\langle t \rangle}{\delta_t} \in \mathbb{R}^{3\times 3} \tag{3.5}$$

Consider an object whose body frame at two consecutive time steps differs by a small rotation ${}^{B}\boldsymbol{R}_{\Delta}$ expressed w.r.t. body frame

$$\boldsymbol{R}_B\langle t+\delta_t \rangle = \boldsymbol{R}_B\langle t \rangle\, {}^{B}\boldsymbol{R}_{\Delta}$$

Now, if we substitute the equation from above and Eq. 3.2 into Eq. 3.5 and rearrange we get

$${}^{B}R_{\Delta} \approx \delta_t \left[{}^{B}\boldsymbol{\omega}\right]_{\times} + \boldsymbol{I}_{3\times 3} \tag{3.6}$$

Eq. 3.6 means that an infinitesimally small rotation can be approximated by the sum of a skew-symmetric matrix and an identity matrix.

This can be demonstrated for example by creating an orthogonal rotation matrix due to 0.001 radian rotation around the x-axis of the original frame:

```
>> rotx(0.001)
ans =
    1.0000         0         0
         0    1.0000   -0.0010
         0    0.0010    1.0000
```

which indeed could also be computed as the sum of a corresponding skew-symmetric matrix and an identity matrix.

From (3.6) it follows that if the angular velocity in the body frame is known we can approximately update the rotation matrix as

$$\boldsymbol{R}_B\langle t+\delta_t\rangle \approx \boldsymbol{R}_B\langle t\rangle + \delta_t \boldsymbol{R}_B\langle t\rangle[\boldsymbol{\omega}]_\times \tag{3.7}$$

Incremental rigid-body motion

To sum up the results from the previous slides we now consider an infinitesimal update of the full 3D-pose:

Suppose that we have two poses ξ_1 and ξ_2 which differ infinitesimally and are related as

$$\xi_2 = \xi_1 \oplus \xi_\Delta$$

We can write the difference transform in homogenous matrix form as

$$\xi_\Delta \sim \boldsymbol{T}_\Delta = \begin{pmatrix} \boldsymbol{R}_\Delta & \boldsymbol{t}_\Delta \\ 0_{1\times3} & 1 \end{pmatrix}$$

where $\boldsymbol{t}_\Delta$ is an incremental displacement and $\boldsymbol{R}_\Delta$ is an incremental rotation matrix (i.e. sum of skew-symmetric matrix and identity matrix)

$$\xi_\Delta \sim \boldsymbol{T}_\Delta = \begin{pmatrix} \left[\boldsymbol{\Delta}_R\right]_\times + \boldsymbol{I}_{3\times3} & \boldsymbol{\Delta}_t \\ \boldsymbol{0}_{1\times3} & 1 \end{pmatrix}$$

Application example: Inertial navigation systems

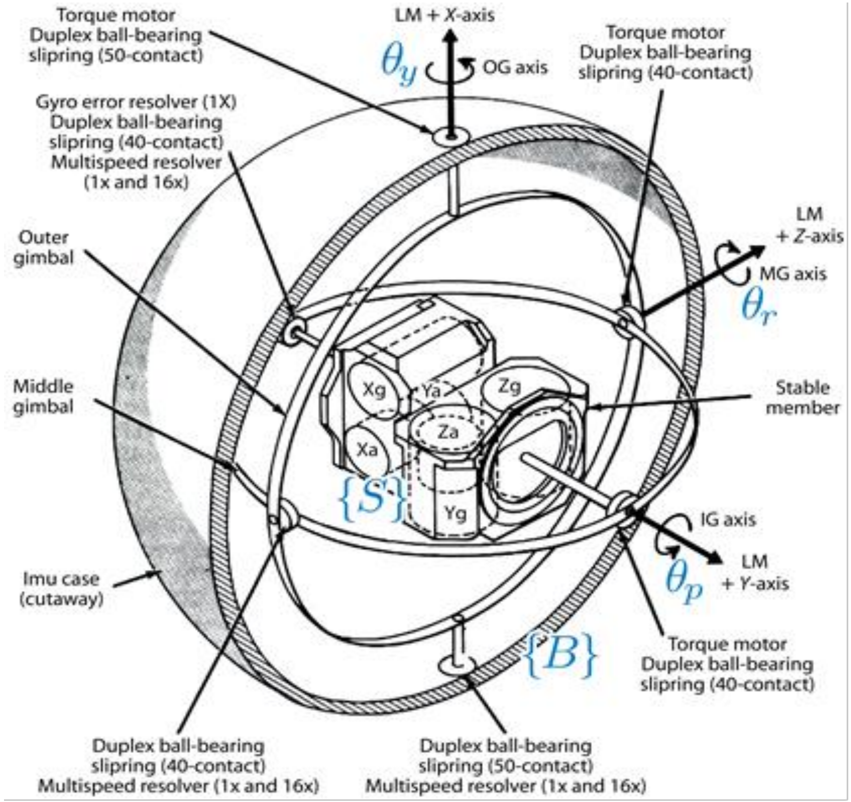
An inertial navigation system (INS) is a “black box” that estimates its velocity, orientation and position with respect to the inertial reference frame (the universe)

It has no external inputs such as radio signals from satellites and this makes it well suited to applications such as submarine, spacecraft and missile guidance.

An inertial navigation system (6-DegreeOfFreedom IMU) works by measuring accelerations and angular velocities and integrating them over time. The sensors measuring accelerations and angular velocities are collectively referred to as an inertial measurement unit (IMU).

If the IMU-unit comprices, in addition to triaxial gyroscopes and accelerometers, also triaxial magnetometers, we refer to it as 9-DOF IMU. By measuring the earth magnetic field, indication about the heading angle can be acquired.

Early inertial navigation systems, (*compare the figure*), used mechanical gimbals to keep the accelerometers at a constant attitude with respect to the stars using a gyro-stabilized platform **->** a (mechanically) very high quality system required **->** very expensive



In a modern strapdown inertial measurement system the acceleration and angular velocity sensors are rigidly attached to the vehicle/device. -> a very small size system, e.g. to fit inside a mobile phone, can be constructed

Determination of the start location of a mobile robot

When using an inertial navigation system (INS) for determining the trajectory of a mobile robot the first step to do before the robot starts to move is to determine the start location of the robot.

To determine the initial position with respect to a map, we can utilize external perception sensors (such as cameras or laser rangefinders) to measure our relative location with respect to natural or artificial landmarks visible to the sensors. Outdoors, on open areas, GPS-satellites are most often used as the artificial landmarks to determine the initial position.

Before to start integrating the position and orientation of the mobile robot from the accelerometer and gyro readings of the inertial measurement unit (IMU) we need an accurate estimate of the initial orientation of the robot w.r.t. the inertial reference frame.

To determine heading of the robot in the inertial reference frame we can use e.g. a compass sensor. The heading angle is described as a rotation around the z-axis of the inertial reference frame, denoted by θ_y i.e. the *yaw*-angle. It will then be followed by two more rotations around y- (*pitch*) and x- axis (roll) to complete the description of the orientation of the robot body frame in 3D.

To determine the tilt angles of the robot body fixed frame {B} w.r.t. inertial reference frame {0} we can utilize the fact that the measured gravitational acceleration vector would be pointing in upward direction (along the z-axis) if the robot was leveled on the xy-plane of the inertial frame of reference:

$$^{0}\boldsymbol{a} = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}$$

First, we describe the orientation of the robot body frame {B} with respect to the inertial reference frame {0} in terms of a three angle sequence:

$$ {}^{0}\xi_{B} = \mathscr{R}_z(\theta_y) \oplus \mathscr{R}_y(\theta_p) \oplus \mathscr{R}_x(\theta_r) $$

that is

$$ {}^{0}\boldsymbol{R}_{B} = \begin{bmatrix} cos\theta_y & -sin\theta_y & 0 \\ sin\theta_y & cos\theta_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_p & 0 & sin\theta_p \\ 0 & 1 & 0 \\ -sin\theta_p & 0 & cos\theta_p \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta_r & -sin\theta_r \\ 0 & sin\theta_r & cos\theta_r \end{bmatrix} $$

Now we can express the gravity vector ${}^{0}\boldsymbol{a}$ (of a leveled platform - i.e. gravity vector pointing along z-axis) w.r.t. frame {B} by multiplying it with an inverse of ${}^{0}\boldsymbol{R}_{B}$

$$ {}^{B}\boldsymbol{a} = \left({}^{0}\boldsymbol{R}_{B} \right)^{-1} {}^{0}\boldsymbol{a} = \left({}^{0}\boldsymbol{R}_{B} \right)^{\boldsymbol{T}} {}^{0}\boldsymbol{a} = \begin{bmatrix} -g sin\theta_p \\ g\ cos\theta_p sin\theta_r \\ g\ cos\theta_p cos\theta_r \end{bmatrix} \quad (3.19) $$

*3-angle sequence that was chosen here gives us equations for the gravity vector **with only** roll and pitch angles to appear in the equations*

The acceleration vector measured by the IMU-sensor in frame {B} is

$$^{B}\boldsymbol{a}^{\#} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$$

By equating this with (3.19) we can solve for the tilt angles (roll, pitch) of the sensor platform

$$\sin\hat{\theta}_p = \frac{-a_x}{g} \qquad (3.20)$$

$$\tan\hat{\theta}_r = \frac{a_y}{a_z}, \quad \theta_p \neq \pm\frac{\pi}{2} \qquad (3.21)$$

In the equations, the hat notation in the pitch θ_p and roll θ_r angles indicates that the <u>values are estimates</u>.

Also, we make the assumption that the sensor platform <u>does not move</u> while registering the acceleration values.

The goal is to estimate the motion of the robot w.r.t. the global inertial frame {0}. The total measured (#) acceleration is due to the gravity and motion:

$$ {}^{0}\boldsymbol{a}^{\#} = {}^{0}\boldsymbol{g} + {}^{0}\boldsymbol{a}_{v} $$

We measure the acceleration w.r.t. the body frame so we have to transform it to {0} for calculating the robot motion w.r.t. {0}

$$ {}^{0}\hat{\boldsymbol{a}}_{v} = {}^{0}\hat{\boldsymbol{R}}_{B}\,{}^{B}\boldsymbol{a}^{\#} - {}^{0}\boldsymbol{g} \qquad (3.22) $$

Note: any error in the rotation matrix leads to an error in the estimated robot acceleration

Integrating w.r.t. time gives the velocity of the mobile robot

$$ {}^{0}\hat{\boldsymbol{v}}_{v}(t) = \int {}^{0}\hat{\boldsymbol{a}}_{v}(t)\,\mathrm{d}t \qquad (3.23) $$

and integrating again gives its position

$$ {}^{0}\hat{\boldsymbol{p}}_{v}(t) = \int {}^{0}\hat{\boldsymbol{v}}_{v}(t)\,\mathrm{d}t \qquad (3.24) $$

Low-cost strapdown IMUs are not very accurate. For the measured values of the the gyroscope, accelerometer or magnetometer signals we should use the following model:

$$x^{\#} = sx + b + \varepsilon$$

where *x* is the unknown true value, *s* is a scale factor, b is an offset/bias and *ε* is random noise.

The scale factor can usually be determined by calibration. The biggest problem is sensor bias which typically varies with time and temperature.

In practice the bias terms need to be estimated regularly because their values are valid only for a short period of time.

The acceleration measured by the IMU is integrated to obtain the velocity of the platform, and integrated again to obtain its position

The three orthogonally mounted gyroscopes measure the components of the angular velocity $\boldsymbol{\omega}$ and use Eq. 3.7 to continuously update the estimated orientation ${}^{0}\boldsymbol{R}_{B}$ of the vehicle's body-fixed frame $\{B\}$ with respect to the environment fixed frame/inertial frame {0}.

Equation 3.7

$$\boldsymbol{R}_B\langle t+\delta_t\rangle \approx \boldsymbol{R}_B\langle t\rangle + \delta_t \boldsymbol{R}_B\langle t\rangle [\boldsymbol{\omega}]_\times \qquad (3.7)$$

is used to numerically integrate changes in pose in order to estimate the orientation of the vehicle

The measured acceleration ${}^{B}\mathbf{a}$ of the vehicle's body frame is rotated into the inertial frame (by using the most recent update of the orientation of the vehicle given by Eq 3.7):

$${}^{0}\boldsymbol{a} = {}^{0}\boldsymbol{R}_B\,{}^{B}\boldsymbol{a}$$

and can then be integrated twice to update the estimate of the vehicle's position in the inertial frame

Practical systems work at high sample rate, typically hundreds of Hertz, and would probably employ higher-order numerical integration techniques rather than the simple rectangular integration of Eq. 3.7

In Eq. 3.7 we added the matrix $\boldsymbol{R}\langle t\rangle\boldsymbol{\delta}_t[\boldsymbol{\omega}]_\mathrm{x}$ to an orthonormal rotation matrix and this is not quite proper – the result will not be an orthonormal matrix

However if the added term is small the result will be close to orthonormal and we can straighten it up

This process is called normalization and enforces the constraints on the elements of an orthonormal matrix.

Note: In an orientation estimation system using Eq. 3.7 the attitude ***R*** should preferably be normalized after each integration step

Rotation matrix normalization involves the following steps where $\boldsymbol{c}_i$ is the i^th column of $\boldsymbol{R}$ (Corke, p.49-50). We first assume that column 3 is correct

$$\boldsymbol{c}_3' = \boldsymbol{c}_3$$

then the first column is made orthogonal to the last two

$$\boldsymbol{c}_1' = \boldsymbol{c}_2 \times \boldsymbol{c}_3'$$

However, the last two columns may not have been orthogonal so

$$\boldsymbol{c}_2' = \boldsymbol{c}_1' \times \boldsymbol{c}_3'$$

Error in the Corke's text book, page 50; the eq. should be $\boldsymbol{c}_2' = \boldsymbol{c}_3' \times \boldsymbol{c}_1'$

Finally the columns are normalized to unit magnitude

$$\boldsymbol{c}_i'' = \frac{\boldsymbol{c}_i'}{|\boldsymbol{c}_i'|}, \quad i = 1 \cdots 3$$

In the Toolbox normalization is implemented by `>> R = trnorm(R);`

As an example, let’s discuss the creation of an orthonormal coordinate frame by means of three reference point vectors (**X1**,**X2**,**X3**) (*which is a similar task to “Rotation matrix normalization”*)

So, the task is to form a right-handed orthogonal coordinate frame according to the following rules:

1. The origin of the coordinate frame should be located at point X1.
2. The x-axis of the orthogonal coordinate frame should be parallel to the vector pointing from X1 to X2 (*marked as* X1X2)
3. The z-axis should be the normal to the plane the basis of which is formed from the X1X2 and X1X3 vectors (*two vectors that span the plane*)
4. The y-axis is determined with right hand-rule to form an orthogonal coordinate frame

The xyz-coordinates of the three points are (expressed as column vectors):

X1 = [3 0 5]'; X2 = [5 2 5]'; X3 = [5 4 5]';

As a solution to the problem, a homogenous 4x4 transformation matrix, that describes the relative position and orientation of the new frame with respect to the reference frame should be formed

To solve the problem (by using MATLAB):

First, initialize the homogenous transformation matrix as the unity matrix:
(you could also use the T=eye(4) command to do the same thing)

```
T = [1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1];
```

and create the vectors X1X2, which will point along new frame x-axis, and X1X3

```
X1X2 = X2 - X1;
X1X3 = X3 - X1;
```

then, calculate the vector which is normal to the plane (determining the z-axis direction of the frame)

```
z_axis = cross(X1X2, X1X3)
```

and now determine the direction of the y-axis with the right hand rule and the x- and z-axis directions calculated above:

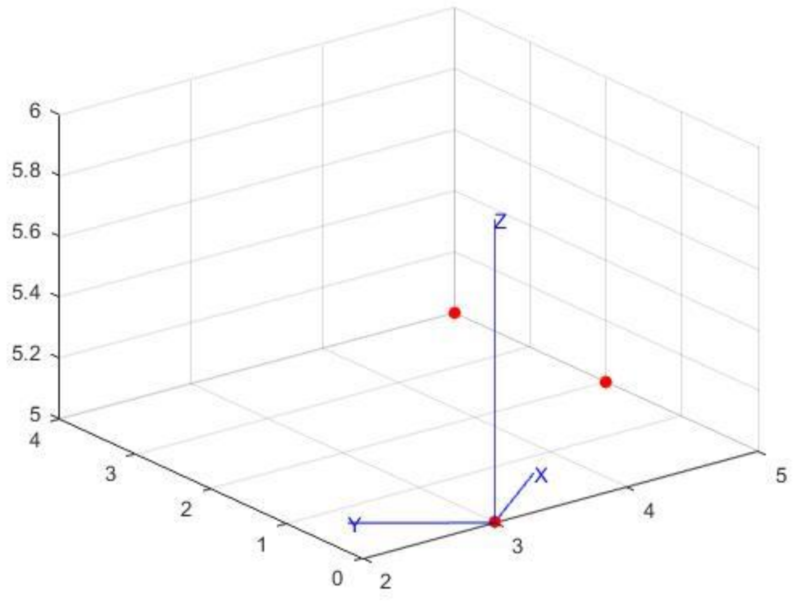
```
y_axis = cross(z_axis, X1X2)
```

and next normalize the frame axis direction vectors to be unit vectors
x_axis = X1X2/norm(X1X2); y_axis = y_axis/norm(y_axis); z_axis = z_axis/norm(z_axis)

and assign the vectors to the corresponding columns of the 4x4 homogenous transformation matrix: T(1:3,1)=x_axis; T(1:3,2)=y_axis; T(1:3,3)=z_axis; T(1:3,4)=X1;

Finally, we can visualize the result by plotting the new frame and the xyz-reference points

```
scatter3(X1(1),X1(2),X1(3),'filled','r');
hold on;
scatter3(X2(1),X2(2),X2(3),'filled','r');
scatter3(X3(1),X3(2),X3(3),'filled','r');
trplot(T, 'color', ‘b');
```



Example problem

1. Integration of rotational motion when using rotation matrix for representing orientation. Imagine that an Inertial Measurement Unit (IMU) has been mounted on board of a mobile robot. The gyroscopes and accelerometers of the IMU are constantly measuring rotational speed and linear acceleration of the robot. The problem consists of two parts.

a) Compute an update of the rotation matrix ***R***, which describes the orientation of the robot w.r.t. the reference (inertial) frame, as a function of the angular speed measurement, $\boldsymbol{\omega}$ acquired from the IMU.

To solve the problem, you can use Eq. (3.7) of Corke's text book. With the equation you can compute an approximation of the rotation matrix at a given time instant δ_t into the future (here 50 ms).

The sum of an orthonormal matrix and another matrix is **not** an orthonormal matrix, but if the added term is small then it will be "close" and we can normalize it. Use the equations, presented on slide 19 to do the normalization. As an answer to the problem, give the numerical forms of the rotation matrix before and after the update.

b) Express the acceleration of the robot, measured w.r.t. the IMU-frame $\boldsymbol{a}_{IMU}$, in the world (inertial) frame $\boldsymbol{a}_{world}$.

The numerical values for the initial rotation matrix ***R*** as well as measured angular velocities $\boldsymbol{\omega}_{IMU}$ and linear accelerations $\boldsymbol{a}_{IMU}$ are as follows:

R = rotx(-30,'deg')*roty(20,'deg')*rotz(40,'deg');

$\boldsymbol{\omega}_{IMU}$ = [0.19 0.06 -0.21]; % gyroscope measurements in radians/sec

$\boldsymbol{a}_{IMU}$=[2.4 1.3 -0.9];   % accelerometer measurements m/s^2

Solution

a) The numerical values for the initial rotation matrix as well as the measured angular velocities and linear accelerations are as follows:

```
R = rotx(-30,'deg')*roty(20,'deg')*rotz(40,'deg'); % initial orientation of IMU

R =

   0.7198   -0.6040    0.3420
   0.4257    0.7733    0.4698
  -0.5483   -0.1926    0.8138


omega_IMU =[0.19 0.06 -0.21];  % gyroscope measurements in radians/sec

a_IMU=[2.4 1.3 -0.9];   % accelerometer measurements m/s2
```

% Now, let’s calculate the updated value of R after the time step 50ms by using Eq.3.7:

```
R = R + R*0.050 * skew(omega_IMU);

R =    0.7252   -0.5932    0.3499
       0.4161    0.7823    0.4638
      -0.5487   -0.1907    0.8140
```

% But now the matrix R is not anymore a proper orthogonal matrix so we
% need to normalize it. Assume that column 3 of R is correct:

% next, make the first column orthogonal to the last two

```
R(1:3,1) = cross(R(1:3,2), R(1:3,3));
```

% and make the second column orthogonal to the third and the first column

```
R(1:3,2) = cross(R(1:3,3), R(1:3,1));
```

```
% and finally normalize the columns to unit magnitude

R(1:3,1) = R(1:3,1)/norm(R(1:3,1));

R(1:3,2) = R(1:3,2)/norm(R(1:3,2));

R(1:3,3) = R(1:3,3)/norm(R(1:3,3));

R =  0.7251   -0.5932    0.3499

     0.4161    0.7822    0.4638

    -0.5488   -0.1907    0.8139
```

R=trnorm(R)

b) Express the acceleration of the robot, measured w.r.t. the IMU-frame a_IMU, in the world (inertial) frame a_world:

```
a_world = R*a_IMU';

a_world = 0.6541

          1.5981

         -2.2975
```

Aalto University
School of Electrical Engineering

Recommended reading:

Peter Corke, *Robotics, Vision and Control, Fundamental Algorithms in MATLAB,* Second Edition, Springer, 2017*,* pages 63-67, 79-88