EN4594 Autonomous Systems
Laboratory Sheet
Practical No. 2

Title: Using MEMS Inertial Measurement Sensors to Determine the Orientation of the Zumo Robot

1. Introduction

Microelectromechanical systems (MEMS) – inertial measurement units (IMU) can be used to estimate the orientation of an attached object (Fig. 1). A 9-DOF IMU consists of a 3-axis accelerometer, a 3-axis gyroscope and a 3-axis magnetometer. The accelerometer measures the *proper acceleration,* which is the acceleration with respect to a free-falling body. The gyroscope reports the angular velocity of the object in all three body axes. The magnetometer measures the Earth's magnetic field and thereby helps finding the heading direction of the object with respect to North direction.
The Zumo 32U4 includes the following on-board inertial sensors that can be used to determine its own orientation: ST LSM303D chip combines an accelerometer sensor and a magnetometer sensor, and ST L3GD20H chip is a gyroscope sensor.

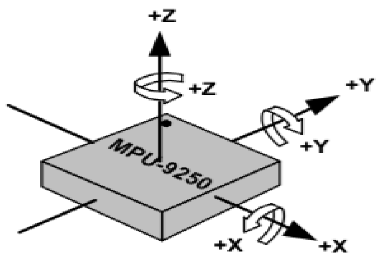

Figure. 1: MEMS IMU sensor for orientation estimation of an object

In this practical you will learn the following:

- Implementation of a Kalman filter assuming a linear Gaussian system
- Estimation of the orientation of a given object using an IMU sensor

2. Theoretical Background

This section describes the theoretical background needed to complete this practical.

2.1. Rotation Matrix

A rotation matrix is a transformation matrix that is used to perform a rotation in Euclidean space. We can use a rotation matrix to give the relative orientation between two rigid bodies. As seen in Fig. 2, the rotation matrix represents one coordinate frame axes (three vectors) in another coordinate frame as its column vectors (eq. (1)).

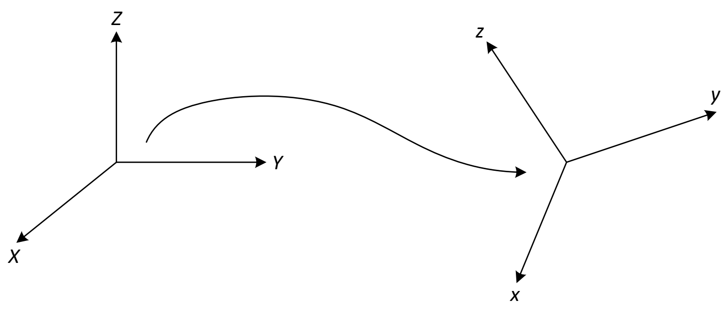

Figure 2: Orientation representation

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{1}$$

where, $\begin{bmatrix} r_{11} \\ r_{21} \\ r_{31} \end{bmatrix}$ = *x* axis expressed in *XYZ* coordinate frame

$\begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix}$ = *y* axis expressed in *XYZ* coordinate frame, and

$\begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \end{bmatrix}$ = *z* axis expressed in *XYZ* coordinate frame

2.2. Kalman Filter

Kalman filter is a probabilistic estimation technique to estimate the state of a system using the system's dynamics and sequential observations in the presence of statistical noise and other inaccuracies. It can be directly applied only for linear Gaussian systems, where both the system dynamic model (eq. (2)) and the observation model (eq. (3)) are linear while the process and measurement noises are zero mean Gaussian.

$$x_t = A_t x_{t-1} + B_t u_{t-1} + \varepsilon_t \tag{2}$$
$$z_t = C_t x_t + \delta_t \tag{3}$$

where, x_t is the *state*, A_t is the *state transition matrix*, B_t is the *control input matrix*, u_t is the control input, z_t is the *observation*, C_t is the *observation matrix*, ε_t is zero mean Gaussian *process noise* with *process noise covariance* R_t and δ_t is zero mean Gaussian *observation noise* with *observation noise covariance* Q_t.

The equations of the Kalman filter are as follows:

- **Prediction**

Predicted state estimate	$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$	(4)
Predicted estimate covariance	$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$	(5)

- **Update**

Innovation	$\bar{z}_t = z_t - C_t \bar{\mu}_t$	(6)
Innovation covariance	$S_t = C_t \bar{\Sigma}_t C_t^T + Q_t$	(7)
Optimal Kalman gain	$K_t = \bar{\Sigma}_t C_t^T S_t^{-1}$	(8)
Update state estimate	$\mu_t = \bar{\mu}_t + K_t \bar{z}_t$	(9)
Updates estimate covariance	$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$	(10)

2.3. Orientation Estimation using Kalman Filter

Each column of the rotation matrix can be considered a state of the system and three Kalman filters can be run in parallel for orientation estimation of the object. we assume that the three states are independent for our convenience.

- **State Transition**

The following equation depicts the change of rotation matrix ${}^E_B R(t)$ in dt time, where E is fixed Earth frame, B is the body frame (Zumo robot) and the orientation change is represented with respect to the fixed frame E. ${}^E_B R(t)$ is the rotation matrix as explained in section 2.1.

$$ {}^E_B R(t+dt) = \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix} {}^E_B R(t) \tag{11} $$

Each column vector of ${}^E_B R(t)$ can be extracted as a state for the Kalman filter and the state transition of each state can be depicted as follows:

$$ {}^E_B \mathrm{x}(t+dt) = \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix} {}^E_B \mathrm{x}(t) \tag{12} $$

$$ {}^E_B \mathrm{y}(t+dt) = \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix} {}^E_B \mathrm{y}(t) \tag{13} $$

$$ {}^E_B \mathrm{z}(t+dt) = \begin{bmatrix} 1 & -d\theta_z & d\theta_y \\ d\theta_z & 1 & -d\theta_x \\ -d\theta_y & d\theta_x & 1 \end{bmatrix} {}^E_B \mathrm{z}(t) \tag{14} $$

where ${}^{E}_{B}\mathrm{x}(t)$, ${}^{E}_{B}\mathrm{y}(t)$ and ${}^{E}_{B}\mathrm{z}(t)$ are the column vectors of ${}^{E}_{B}R(t)$. In order to construct the state transition matrix, the gyro readings can be used as follows:

$$d\theta_x = \omega_x dt \tag{15}$$
$$d\theta_y = \omega_y dt \tag{16}$$
$$d\theta_z = \omega_z dt \tag{17}$$

- **Observations**

1) Roll, pitch measurements
The accelerometer sensor can be utilized to generate roll and pitch measurements of the object considered. It can be shown that the rotation matrix for successive roll ψ, pitch θ and yaw ϕ angles (in the given order) about the fixed Earth E frame is given by:

$$ {}^{E}_{B}R = \begin{bmatrix} cos\phi cos\theta & -sin\phi cos\psi + cos\phi sin\theta sin\psi & sin\phi sin\psi + cos\phi sin\theta cos\psi \\ sin\phi cos\theta & cos\phi cos\psi + sin\phi sin\theta sin\psi & -cos\phi sin\psi + sin\phi sin\theta cos\psi \\ -sin\theta & cos\theta sin\psi & cos\theta cos\psi \end{bmatrix} \tag{18}$$

Using eq. (18), Earth's gravitational field vector g can be transformed to the body frame as follows:

$$ {}^{B}g = {}^{B}_{E}R\, {}^{E}g \tag{19}$$

Since,

$$ {}^{B}_{E}R = ({}^{E}_{B}R)^{-1} = ({}^{E}_{B}R)^{T} \tag{20}$$

eq. (19) can be written as,

$$\begin{bmatrix} {}^{B}g_x \\ {}^{B}g_y \\ {}^{B}g_z \end{bmatrix} = ({}^{E}_{B}R)^{T} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{21}$$

Note that IMU accelerometer outputs *proper acceleration* (acceleration w.r.t a falling body) and therefore, we have used $[0 \quad 0 \quad 1]^T$ instead of $[0 \quad 0 \quad -1]^T$. Consequently, eq. (21) becomes,

$$\begin{bmatrix} {}^{B}g_x \\ {}^{B}g_y \\ {}^{B}g_z \end{bmatrix} = \begin{bmatrix} -sin\theta \\ cos\theta sin\psi \\ cos\theta cos\psi \end{bmatrix} \tag{22}$$

From eq. (22), roll ψ can be found by:

$$\psi = tan^{-1}\left(\frac{{}^{B}g_y}{{}^{B}g_z}\right) \tag{23}$$

Also, from eq. (22), pitch θ can be found by:

$$\theta = tan^{-1}\left(\frac{-{}^{B}g_x}{\sqrt{{}^{B}g_y{}^{2} + {}^{B}g_z{}^{2}}}\right) \tag{24}$$

2) Yaw measurements

Using the magnetometer sensor and the results obtained for roll and pitch measurements, the yaw measurements can be obtained. Yaw ϕ can be defined as the angle of the projected x-axis of the body frame on the ground with respect to magnetic North direction in positive anticlockwise sense as illustrated in Fig. 3.

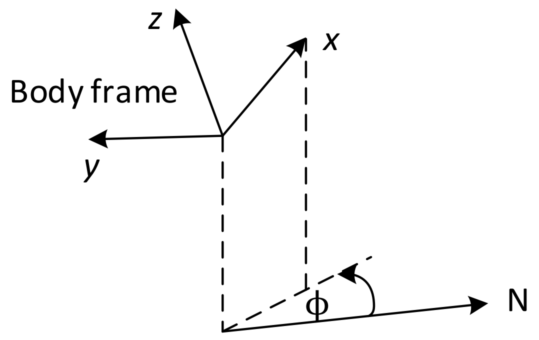

Figure 3: Yaw measurement

In order to obtain the yaw ϕ measurement, we transform the magnetometer reading ${}^{B}M$ to a frame whose XY plane is coplanar with the Earth XY plane (frame-1), as illustrated in Fig. 4.

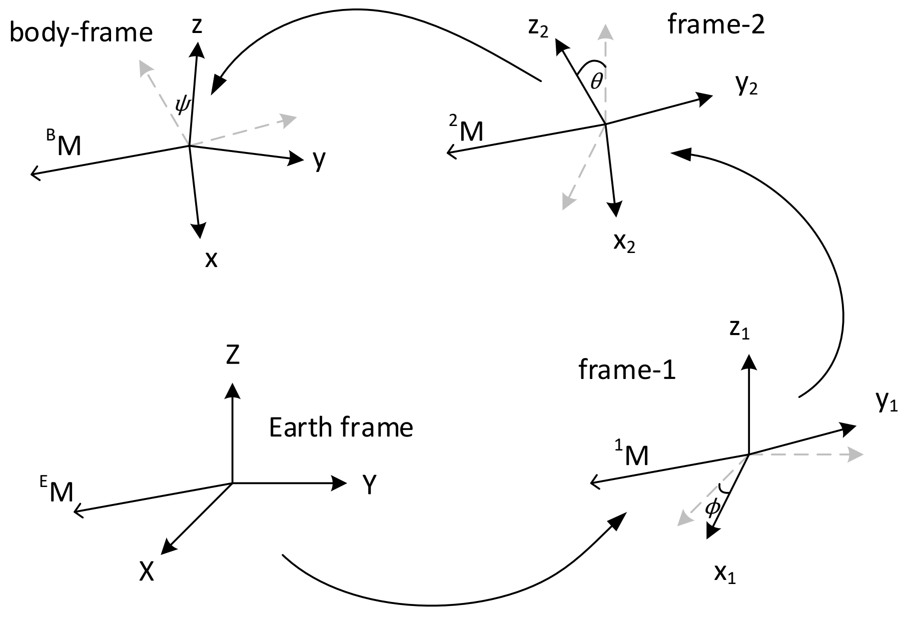

Figure 4. Transforming magnetometer readings to different frames

$$ {}^{1}M = {}^{1}_{B}R\,{}^{B}M \tag{25}$$

${}^{1}_{B}R$ can be generated by basic rotation matrices as follows:

$$ {}^{1}_{B}R = {}^{1}_{2}R \cdot {}^{2}_{B}R \tag{26}$$

Thus, eq. (25) becomes

$$ {}^{1}M = {}^{1}_{2}R\,{}^{2}_{B}R\,{}^{B}M \tag{27}$$

where,

$$ {}^{1}_{2}R = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \qquad {}^{2}_{B}R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix} \tag{28}$$

and, ${}^{B}M$ is the magnetometer reading:

$$ {}^{B}M = {}^{B}\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} \tag{29}$$

Therefore, from eq. (27) we get,

$$ {}^{1}\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix} {}^{B}\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} \tag{30}$$

To obtain, magnetic North direction we project ${}^{1}M$ on to the *xy* plane by letting ${}^{1}m_z = 0$. As a result, we get

$$ {}^{1}_{proj}\begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} {}^{B}m_x\cos\theta + {}^{B}m_y\sin\theta\sin\psi + {}^{B}m_z\sin\theta\cos\psi \\ {}^{B}m_y\cos\psi - {}^{B}m_z\sin\psi \\ 0 \end{bmatrix} \tag{31}$$

where, we can replace pitch θ and roll ψ angles by the readings obtained previously. Since this projected ${}^{1}_{proj}M$ gives North direction in frame-1, the yaw ϕ measurement is given by:

$$ \phi = -tan^{-1}\left(\frac{{}^{B}m_y\cos\psi - {}^{B}m_z\sin\psi}{{}^{B}m_x\cos\theta + {}^{B}m_y\sin\theta\sin\psi + {}^{B}m_z\sin\theta\cos\psi}\right) \tag{32}$$

3) Inclusion of Observations
   We use the obtained roll, pitch and yaw observations to generate ${}^{E}_{B}R$ matrix as given in eq. (18). Finally, we use the columns of this matrix to be compared with the predicted rotation matrix in eq. (11).

2.4. Renormalization

Rotation matrix needs to comply orthogonality conditions; i.e., its column vectors should be orthogonal to each other and should be unit vectors. State transition explained in section 2.3 will gradually accumulate errors and will violate these orthogonality conditions. Therefore, we need to normalize the column vectors (or row vectors) of the rotation matrix to enforce its orthogonality.

Orthogonality implies that the dot product of the column vectors (also row vectors) of the rotation matrix is zero. Any deviation from zero will let us know the *error* (eq. (33)).

$$x = \begin{bmatrix} r_{11} \\ r_{21} \\ r_{31} \end{bmatrix} \quad y = \begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix} \quad z = \begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \end{bmatrix}$$

$$error = x \cdot y = x^T y = \begin{bmatrix} r_{11} & r_{21} & r_{31} \end{bmatrix} \begin{bmatrix} r_{12} \\ r_{22} \\ r_{32} \end{bmatrix} \tag{33}$$

Among many different ways of correcting the error, the method that will be used is to assign half the error to each column vector and approximately rotate them in the opposite direction.

The adjusted vectors are then given by:

$$x_{orth} = x - \frac{error}{2} y \tag{34}$$

$$y_{orth} = y - \frac{error}{2} x \tag{35}$$

The remaining vector is obtained by the cross product of the adjusted vectors in eq. (34) and (35).

$$z_{orth} = x_{orth} \times y_{orth} \tag{36}$$

After orthogonalizing the rotation matrix column vectors, they should be normalized.

$$x_{norm} = \frac{x}{|x|}, \; y_{norm} = \frac{y}{|y|}, \; z_{norm} = \frac{z}{|z|} \tag{37}$$

For embedded systems, the following equations approximate the normalization using 1st order Taylor expansion.

$$x_{norm} = \frac{1}{2}(3 - x_{orth} \cdot x_{orth}) x_{orth} \tag{38}$$

$$y_{norm} = \frac{1}{2}(3 - y_{orth} \cdot y_{orth}) y_{orth} \tag{39}$$

$$z_{norm} = \frac{1}{2}(3 - z_{orth} \cdot z_{orth}) z_{orth} \tag{40}$$

3. Procedure

This section describes the necessary procedure for Kalman filter based orientation estimation.

3.1. Folders

- Copy the following folders into your ROS workspace `src` folder:
```
zumo ------- zumo_description
       |---- zumo_imu_kf
```

- Update the following existing folder in your ROS workspace `src` folder:
```
zumo ------- zumo_launch
```

3.2. Configurations

- Copy the `zumo_imu.rviz` file into `.rviz2` hidden folder in the home directory.

3.3. Install Other Required Libraries

- Xacro (XML Macros)
  Xacro is a powerful tool for defining robot models in URDF (Unified Robot Description Format) using XML macros.
  Open a new terminal and type:
```
sudo apt-get update
sudo apt install ros-humble-xacro
```

3.4. Kalman Filter Implementation

Implement the following in `zumo/zumo_imu_kf/src/zumo_imu_kf.cpp` file:

- Roll and pitch angle calculation in the `ZumoImuKF::TransformAccRawData` function as explained in section 2.3.
- Yaw angle calculation in the `ZumoImuKF::TransformMagRawData` function as explained in section 2.3.
- Perform renormalization in the `ZumoImuKF::Normalize()` function as explained in section 2.4.
- Kalman filter prediction step in the `ZumoImuKF::KalmanPredict()` function as explained in section 2.2.
- Kalman filter update step in the `ZumoImuKF::KalmanUpdate()` function as explained in section 2.2.

3.5. Build the Nodes

Open a new terminal and navigate to your ROS workspace.
e.g. `cd ~/ros2_ws/`
To build the workspace use the following command:
```
colcon build --symlink-install
```

If you encounter any compilation errors, contact the instructor to resolve them.

3.6. Zumo Connection

- Connect the Zumo 32U4 robot to the computer using the USB A to Micro-B cable.
- To check the USB connection, open a new terminal and type:
  ```
  cd /dev
  ls
  ```
  Check whether `ttyACM0` (`ttyACM1, ttyUSB0` or something similar) exists after connecting the Zumo robot.
- Give permission to access the device.
  ```
  sudo chmod a+rw /dev/ttyACM0
  ```

3.7. Running ROS Nodes

- Follow the instructions given in section 4.1 to start the Zumo ROS interface.
- Follow the instructions given in section 4.2 to observe Zumo sensor data.
- Upload robot structure for visualization purposes as described in section 4.3.
- Copy the accelerometer and magnetometer calibration values generated through calibration experiments into `zumo_imu_kf.launch` file in `zumo_launch` package.
- Run the Kalman filter as described in section 4.4.
- Follow the instructions given in section 4.5 to observe the orientation of the robot estimated by the Kalman filter.
- Comment out the `KalmanUpdate();` line in `ZumoImuKF::ZumoSensorCb` function and observe what happens in the `rviz2` visualization.

4. ROS Reference

This section describes ROS commands to startup ROS nodes.

4.1. Zumo Interface

- Open a new terminal and type:
  ```
  ros2 launch zumo_launch zumo_serial_node.launch
  ```
  This command will establish serial communication with the ZUMO robot.

  Note: If the serial port is different from `/dev/ttyACM0`, you need to change `zumo_serial_port` parameter in `zumo_serial_node.launch` file in `zumo_launch` package. You may use IDE opened in the ROS workspace.
  Open up a new terminal and type:
  ```
  cd ros2_ws/
  code .
  ```

4.2. Observing Zumo Sensor Data

- Open up a new terminal and type:
  ```
  ros2 topic echo /zumo/zumo_sensors
  ```
  This command will echo sensor data in the terminal. You may rotate the track wheels or change the orientation of the robot to observe changes in sensor data.

4.3. Uploading Zumo Description

- Open up a new terminal and type:
  ```
  ros2 launch zumo_launch zumo_startup.launch.py
  ```
  This command will upload the Zumo robot structure for visualization purposes.

4.4. Starting IMU KF Node

- Open up a new terminal and type:
  ```
  ros2 launch zumo_launch zumo_imu_kf.launch
  ```

4.5. ROS Visualization

- Open up a new terminal and type:
  ```
  ros2 run rviz2 rviz2 -d ~/.rviz2/zumo_imu.rviz
  ```

Note

To stop any node, press `Ctrl+c` on the terminal.