**Implementing a ROS node on Android using IMU + GPS**

Contents

[Description: 1](#_Toc30689539)

[Implementation Notes (Timeline): 1](#_Toc30689540)

[To Do: 1](#_Toc30689541)

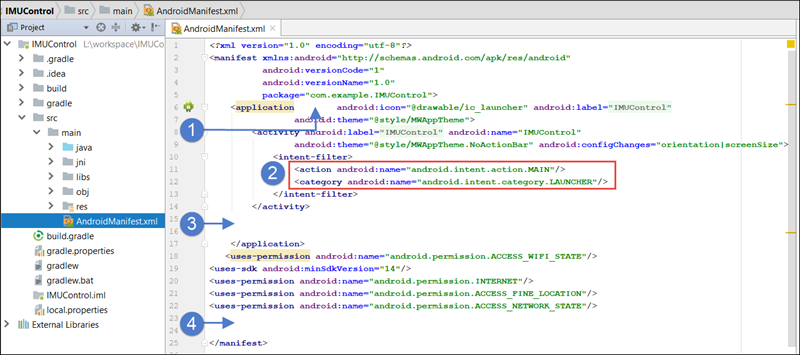
[Troubleshooting: 1](#_Toc30689542)

# Description

1. Matlab has a built-in example named “Visualize Sensor Data from Android Device in RViz”. However, it does not have the ROS nodes support in the form of built-in blocks. So, ROS functionality has to be implemented through ROSJava core in Android Studio. Matlab instructions didn’t lead to a successful compile. Followed the original author’s modified project on Mathworks’ file exchange.

# Implementation Notes (Timeline-revisit to edit previous steps)

1. After deploying the app through Simulink, import the project in Android Studio. Rebuild to make sure it compiles here.
2. Copy the following files to the java source code folder (e.g. src\main\java\com\example\ROS\_Android\_Sensors)
   1. ROSMainActivity.java
   2. PublishNodeIMU.java
   3. PublishNodeGPS.java
   4. PublishNodeTemp.java
   5. PublishNodePress.java
   6. PublishNodeMag.java
   7. MainActivity.java
   8. MWNodeMainExecutorListener.java
   9. MWNodeMainExecutorService.java
   10. MWNodeMainExecutorServiceListener
3. Edit the package path at top of all these copied files i.e. com.example.ROS\_Android\_Sensors;
4. Copy the main.xml file to src > main > res > layout folder.
5. Copy the mwstyle.xml file to src > main > res > values folder.
6. Copy ‘build.gradle’ and replace the one already in the project folder.
7. Copy ‘gradle.properties’ and replace the one already in the project folder.
8. Copy ‘gradle-wrapper.properties’ and replace the one already in the project folder.
9. The above three steps affect the Gradle version, min sdk, target sdk and dependencies. These play an important part in the compilation. Change these later if necessary to target some devices.
10. Edit AndroidManifest:



1. Add the following immediately after ‘<application’

xmlns:tools="http://schemas.android.com/tools"

tools:replace="android:icon"

1. Replace the action and category sections

<category android:name="android.intent.category.DEFAULT" />

<action android:name="com.android.rosupport.START\_MODEL\_IMU"/>

1. Add the activity and service sections immediately after the activity section.

<activity android:label="@string/app\_name" android:name=".MainActivity"

android:theme="@style/MWAppTheme" android:configChanges="orientation|screenSize">

<intent-filter>

<action android:name="android.intent.action.MAIN"/>>

<category android:name="android.intent.category.LAUNCHER"/>

</intent-filter>

</activity>

<service android:name=".MWNodeMainExecutorService"

tools:ignore="ExportedService">

<intent-filter>

<action android:name="org.ros.android.NodeMainExecutorService"/>

</intent-filter>

</service>

1. Add permissions

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>

<uses-permission android:name="android.permission.INTERNET"/>

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"/>

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE"/>

<uses-permission android:name="android.permission.WAKE\_LOCK"/>

<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW"/>

1. Replace AppCompactActivity with ROSMainActivity in the main file ROS\_Android\_Sensors.java
2. Copy all the imports to the main file.
3. Add a variable for each of the ROS publisher
   1. PublishNodeIMU pn\_imu;
   2. PublishNodeGPS pn\_gps;
   3. PublishNodeMag pn\_mag;
   4. PublishNodeGPS pn\_temp;
   5. PublishNodeGPS pn\_press;
4. Add the ROSPublish and init methods definitions.

public void ROSPublishFast(float[] data)

{

if(pn\_imu != null)

pn\_imu.publishData(data);

}

public void ROSPublishSlow(float[] data)

{

if(pn\_gps != null)

pn\_gps.publishData(data);

}

protected void init(NodeMainExecutor nodeMainExecutor) {

pn\_imu = new PublishNode("/android/imu","sensor\_msgs/Imu");

NodeConfiguration nodeConfiguration = NodeConfiguration.newPublic(getRosHostname());

nodeConfiguration.setMasterUri(getMasterUri());

nodeConfiguration.setNodeName("android\_sensors\_driver\_imu");

nodeMainExecutor.execute(pn\_imu, nodeConfiguration);

pn\_gps = new PublishNodeGPS("/android/gps","sensor\_msgs/NavSatFix");

NodeConfiguration nodeConfiguration2 = NodeConfiguration.newPublic(getRosHostname());

nodeConfiguration2.setMasterUri(getMasterUri());

nodeConfiguration2.setNodeName("android\_sensors\_driver\_nav\_sat\_fix");

nodeMainExecutor.execute(pn\_gps, nodeConfiguration2);

};

1. Sync and compile
2. AppFragment.java, InfoFragment.java, CameraScopeFragment.java and some other files may have a problem in import libs due to conversion from appcompat to androidx. Similarly, layout files (xml) may have android components as well. Replace the relevant code from the earlier compiled project and edit.
3. **If there is no change in interfaces e.g. screed displays etc. and only the functionality is changed, then no need to re-import the whole project in Android Studio. Simply, copy and past the lib file in “src\main\libs\armeabi-v7a”.**
4. **In the function, getOrientationData() in Android Studio, comment the code converting angles to degrees. Simulink model assumes radians and hence gives almost noise if not corrected.**
5. **In GPSHandler.java, change the value of the variable ‘MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES’ to ‘1’ and comment the code which takes updates from the network. Otherwise, the GPS updates are too few.**

# To Do

1. To make the app keep working while in locked state. Dim lock state seems to work but not in full locked case.

# Troubleshooting