Intel® Edison Robotics Sample Guide

Introduction

This example uses sensors from the Starter & Robotics kits mounted on a Seeed* 4WD Hercules Mobile Platform to illustrate the concept of a smart vehicle. Other rover or robot frames are also suitable for the project as long as all the sensors are available. The robot is controlled via string commands with the idea of using the process as the child of another front-end service (e.g. webserver) that is running on the Intel® Edison.

The sensors used are:

- I2C Motor Shield and DC Motors as the driving components.
- IR Distance Interrupters for obstacle detection and collision avoidance.
- A digital compass for determining orientation and navigation assistance.
- An I2C LCD for displaying information and telemetry data.
- A Voltage Divider for monitor the battery level.

The source code for the Robotics sample is located in the following repository:

https://github.com/intel-iot-devkit/iot-devkit-samples/tree/master/kits/robotics

Prerequisites

Complete the Getting Started section on the Intel Developer Zone:

https://software.intel.com/en-us/iot/library/edison-getting-started

Even though it is possible to compile and link the source code for this demo directly on the Intel® Edison using the integrated GNU Compiler, it is recommended to use the Eclipse* IDE as this robotics sample is already a part of the Eclipse* project templates. The Getting Started guide includes instructions on how to install Eclipse*.

Due to the nature of this project and the involved components, you will also need a straight head precision screwdriver to attach the I2C Motor Shield terminals. A full precision tool set should make any assembly possible regardless of the rover platform used. Insulation tape, long nose pliers, extra wires and wire strippers are also useful things to have. Lastly, some zip ties and/or twist ties will help keep wire clutter to a minimum.

Hardware Setup

The project uses the following parts and sensors:

- Intel® Edison kit with Arduino* breakout board and a USB to micro USB cable
- Grove* – Base Shield (SLD01099P)
- Grove* - LCD RGB Backlight (811004001)
- Grove* - 3-Axis Digital Compass (SEN12753P)
- Grove* - I2C Motor Driver (ROB72212P)
- Grove* - Voltage Divider (POW05161P)
- 4x Grove* - IR Distance Interrupter (SEN09281P)
- 4x Seeed* 25GA35 DC Motors (RK-370C-3080EP)

Intel® Edison Kit	Base Shield	LCD	Compass
Motor Driver	Voltage Divider	Distance Interrupter	DC Motors

Additional parts include:

- SeeedStudio* 4WD Hercules Mobile Platform Kit (KIT06071P)
- 7.4V (2-cell) LiPo Battery

Hercules Mobile Platform	7.4V LiPo Battery

These additional parts will be shown and used throughout this demo, but they are not project specific and could be replaced by alternatives. The code sample will also work for 2WD rovers.

While assembling the parts and testing them, the LiPo battery can be replaced safely with a variable power supply, but make sure your robot is secure and not touching the ground (e.g. leave the wheels off until everything else is in place and tested).

With a 2-cell 7.4V LiPo battery, the rover will power down on its own before the battery is fully discharged as it drops below the minimum rated voltage (~7V) for the Intel® Edison board. For additional power and running time, a 3-cell 11.2V LiPo battery can be used. In this case however, the battery level needs to be monitored and the Intel® Edison needs to be shut down manually when the battery is low, before battery damage occurs. As a general rule of thumb, the cut-off voltage for a 3-cell LiPo is at 9V under load. The provided code does not include a threshold based shut down feature.

Table 1 Sensor connections to the Grove Base Shield*

Sensor	Connection	Notes
Grove* - LCD RGB Backlight	I2C	
Grove* - 3-Axis Digital Compass	I2C	See Known Limitations
Grove* - I2C Motor Driver	I2C	Connect right side motors to M1 and left side motors to M2
Grove* - Voltage Divider	A0	Input from power supply
Grove* - IR Distance Interrupters	D2	Front Left Sensor
	D4	Front Right Sensor
	D6	Rear Left Sensor
	D8	Rear Right Sensor

The sensor connections to the Grove* Base Shield are listed in Table 1. A snapshot of the Grove* Base shield with the different connector types highlighted is also given in Figure 1. For the I2C connections, it does not matter which slots are used since it is a bus.

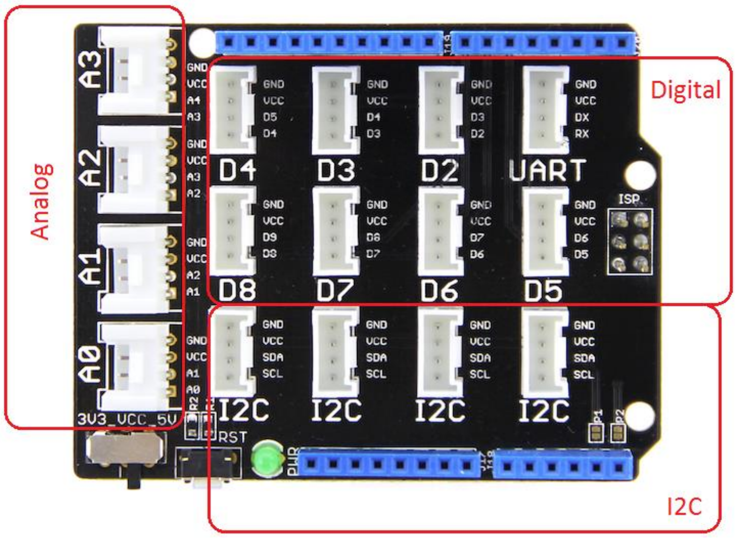

Figure 1 - Grove Base Shield and connection types.*

If you're using the SeeedStudio* Hercules Mobile Platform, there is a guide available at the following link that explains how it should be assembled:

http://www.seeedstudio.com/recipe/216-edison-4wd-auto-robotic-platform.html

Other connections and sensor placement considerations are discussed and illustrated for the remainder of the section, starting with some additional figures showing the process of building the SeeedStudio* Hercules Mobile Platform at various stages. Use these figures in conjunction with the recipe provided by SeeedStudio* to understand how the sensors from the Robotics Kit and the rover frame are utilized together.

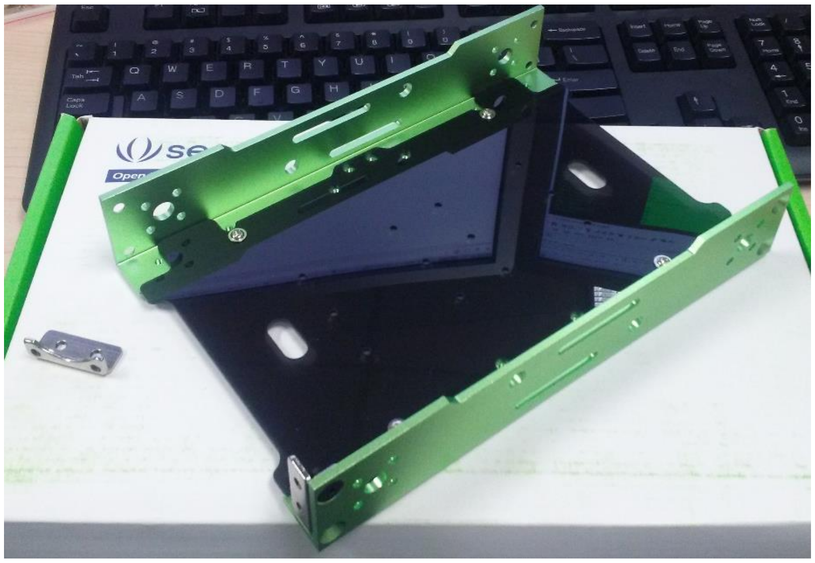

Figure 2 - Lower rover frame assembled.

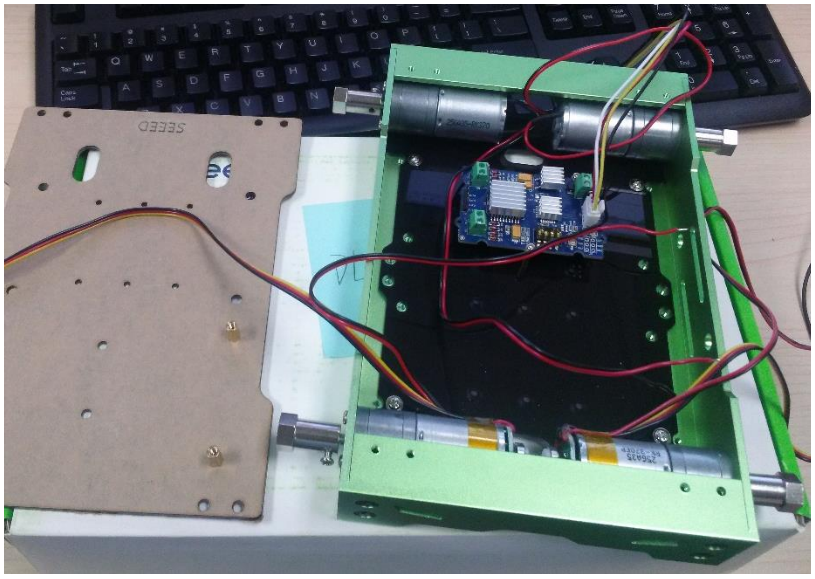

Figure 3 - DC Motors and I2C Motor Shield added to the frame.

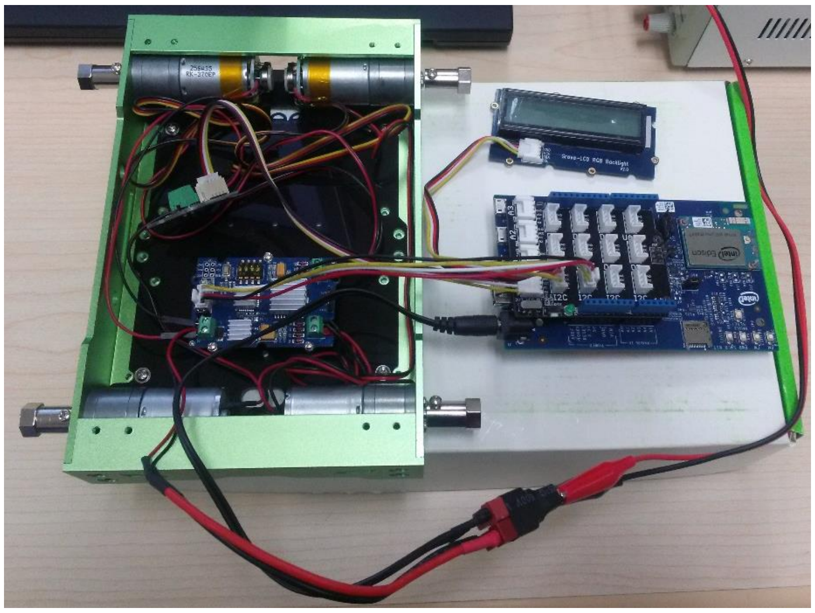

Figure 4 - Motors connected, Intel® Edison, LCD and voltage divider in place for testing.

Note that the Grove* 3-Axis Digital Compass is not shown due to Known Limitations. Also keep in mind that in Figure 4 the front of the rover is facing down. In order to ensure compatibility with the provided code, the DC motors on the right side need to be connected to Channel 1 (M1) on the motor driver and the left side motors to Channel 2 (M2). The power terminal block on the I2C motor driver can connect both the power source and the voltage divider input, used to monitor the battery level, to the circuit.

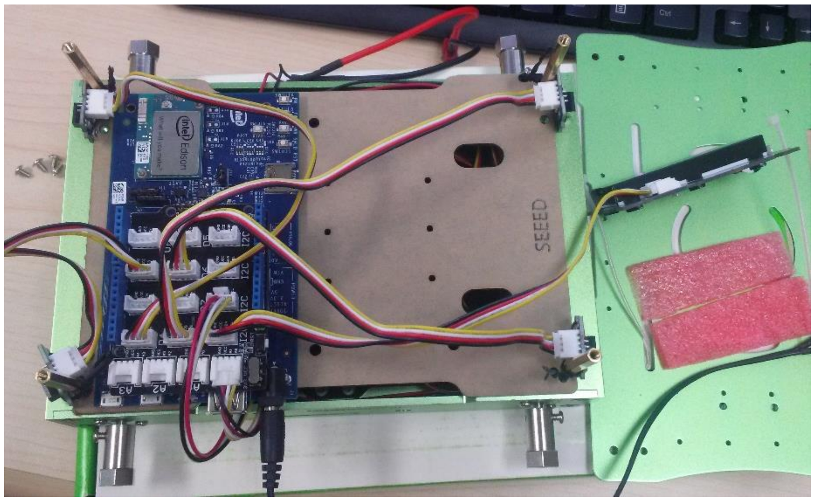

Figure 5 - IR Distance interrupters attached to the top platform struts.

The infrared distance interrupters are used to stop the robot and prevent collisions with static or dynamic obstacles. Positioning and calibration of these sensors is essential for maximizing their usability in this application. Detection range of the sensors can be adjusted with the R8 potentiometer located on the back side of the sensor (same side as the 4 pin connector). Common practice is to place the distance sensors on the corners of the robot, see Figure 5, and have them tilted slightly to the side to cover the wide open-wheels. Attaching them to the hexagonal distance holders between the base of the rover and the top plate is recommended because it will be possible to rotate them to the desired angle and test their coverage before fastening them into place.

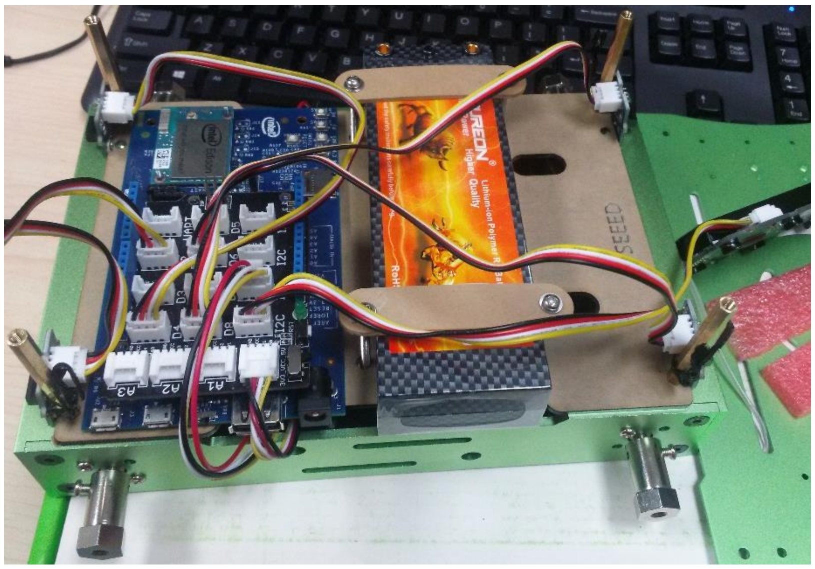

Figure 6 - Top level complete with battery installed.

Once this stage is complete, simply mount the top platform, wheels and secure the cables with some zip or twist ties and the rover will be ready.

Known Limitations

The Grove* 3-Axis Digital Compass is not compatible with the Intel® Edison Arduino board when the board is in 5V mode. To work around this issue, the recommended solution is to use an external voltage level shifter (also known as level translator/converter). Some example products that can be used are:

- Adafruit* 4-channel I2C-safe Bi-directional Logic Level Converter - BSS138
- SparkFun* Logic Level Converter - Bi-Directional
- SparkFun Level Translator Breakout - PCA9306

If the Intel® Edison Mini-Breakout board is used instead, then this limitation does not apply.

Once the front left wheel is in position, the micro USB ports on the Intel® Edison board will be obstructed, thus make sure the board has WiFi connectivity beforehand.

Running and Using the Code

The preferred method to run the application is using the Eclipse* IDE, as the project is available as a template with the rest of the Intel® IoT Developer Kit projects. Find the Eclipse* IDE for download here:

https://software.intel.com/iot/downloads#ide

By the time of reading this document, the sample might have been updated with additional features. In this case, update the Eclipse* sample project with the new code available in the Github* iot-devkit-samples repository. If new sensors were added, make sure you also update the linker flags in the project settings either manually or by using the IoT Sensor Support tab. As a reminder, all the information required to become familiar with the Intel® Developer Kit version of the Eclipse* IDE can be found in the Getting Started Guide.

When using Eclipse*, the option to compile and run the code on the target device, in this case the Intel® Edison, is readily available. This can be used for testing purposes, but recognize that the binary will be saved under a temporary folder and will be removed after the device reboots. Thus it is recommended that you copy the binary manually to a location on the Intel® Edison where it can be accessed later so that a different process running on the board automatically on a reboot can load this control application as needed.

The application is multithreaded, with separate threads for the separate sensors. The main thread is responsible for receiving, processing commands and sending them to the motor driver. A mutex is in place to avoid contention on writing to the LCD display from the compass and the voltage divider threads while a fourth and final thread handles the infrared interrupters. Any of the threads can be disabled by editing the code if the corresponding sensor functionality is not desired, that is, except for the main thread.

Table 2 Available commands for rover control.

Command	Notes
fwd [0-255]	Move rover forward at set speed. Speed value is mandatory.
rev [0-255]	Move rover in reverse at set speed. Speed value is mandatory.
left [0-255]	Turn left in place at set speed. Speed value is mandatory.
right [0-255]	Turn right in place at set speed. Speed value is mandatory.
stop	Stops all movement. Equivalent to any of the previous commands with speed set to 0.

Table 2 lists the commands available as input to control the rover and its movements. A basic filter is in place that will not accept commands other than the ones listed above (in the exact format specified). That is, after every directional command, an integer between 0 and 255 needs to be added signifying the speed. The stop command can be used without the numerical argument, and is also executed by default when an invalid input is received or when any of the infrared interrupters is triggered.

A global variable named *batteryThreshold* is defined and used to indicate a low battery state by comparison with values read from the voltage divider. The value is hardcoded and set by default to 7.2f. The number represents voltage and can be changed to any desired value before compiling the code. Please refer to the Hardware Setup section for more information about minimum operating voltages and LiPo battery considerations.

In order to exit the control application cleanly, you will have to send a SIGINT (Ctrl+C on Linux* systems) followed by an EOF (Ctrl+D on Linux* systems). Since the main loop is waiting on input, it will not process the interrupt signal until the input stream is closed, thus these two separate steps are required. This should be rather trivial to achieve when the control application is running as a child process of another process. While it can be used as a standalone program, the intended use for the provided code is to be launched by a parent server process hosted on the Intel® Edison. The server would then be responsible for both rover control and handling communication between the board and a remote device.

Possible Extensions for this Sample

The most straightforward extension to a 4WD rover would be to allow for remote control and operation of the vehicle through a nicely crafted user interface. Using an Intel® Edison board, this becomes very intuitive as the integrated WiFi is well suited for such an application. A built-in u.fl connector also allows for external antennas to be connected. When coupled with a powerful access point or base station, this will allow for bidirectional communication over extended ranges allowing you to send commands to the rover and receive feedback from the onboard sensors.

Live video streaming in compressed formats is also possible given the processing power of the Intel® Atom core, and some example projects, most notably *edi-cam*, can show how to setup a video camera on the Intel® Edison with minimal effort.

Note that the *edi-cam* project does not provide audio streaming and uses a Node.js server and WebSockets for streaming the video. The encoder is ffmpeg and decoding is done on the client side with jsmpeg using MPEG-1 format for the video. UVC drivers are already bundled with the latest Intel® Edison Yocto images, thus enabling the supported video cameras as soon as they are plugged into the board.

Additionally, gstreamer and alsa drivers are available in the Intel® Edison repositories. When installed, they should enable compiling video and audio streaming applications using the newer H264 and MPEG-4 codecs.

Another interesting idea involves the addition of a GPS sensor for tracking the location of the rover and potential automated navigation when combined with the compass. An example of how to use the GPS sensor is given in our transportation sample.

For those using the Intel® Edison Mini-Breakout for this project, a 5V low dropout voltage regulator will help powering the 5V sensors since the Mini-Breakout design does not provide a steady 5V supply by default.

Please follow the intel-devkit-samples repository for examples and code files showcasing some of these extensions as they are made available.

Other Considerations and Disclaimer

Unlike most of the other samples provided within the Intel® IoT Developer Kit repository, this particular sample makes use of an additional power supply and/or a Lithium battery, therefore adding the risk of electric shock or fire resulting from damage in case of the battery. The H-Bridge chip on the motor driver will get hot under continued use especially at full power and under load. Always double check the polarities of the connections as inversions are the culprit behind hardware damage most of the time. Mind the moving parts and driving the rover once assembled.

Intel Corporation should not be held responsible in the event of misuse, malfunction, damaged hardware/property or personal injury from the assembly and use of this sample project or the included code.