Report on Implementation of VDetect Tool

DIT-ASTEC Wireless Sensor Networks Project
Statistical Signal Processing Lab
Department of ECE
Indian Institute of Science
Prateek G V and K V S Hari

June 21, 2012

1 About VDetect Tool

In the process of working in this project, we realized that the most tedious part of the project was to collect magnetometer data for different vehicles. Based on the analysis done, we decided to develop an in-house tool which would make the process of collection of magnetometer data an easier task. The main functionality of this tool includes collection of data, plotting of magnetometer data and the count of the vehicles detected.

In this report, we will look at how to use this tool, some of the basic functionality and future work that can be done to improve this tool.

2 Installing the wxPython, numpy and gnuplot

The tool runs on a set of open-source softwares such as wxPython (a GUI based python interactive tool), numpy (additional support tool used by python to run high end mathematical applications) and gnuplot (an open-source tool used for plotting data).

2.1 For Ubuntu Users

2.1.1 Installing wxPython

- sudo apt-get install curl
- Go to http://wiki.wxpython.org/InstallingOnUbuntuOrDebian and follow the steps. Install natty deb file and modify accordingly.

2.1.2 Installing numpy

- sudo apt-get install python-numpy python-scipy

2.1.3 Installing gnuplot - Method 1

- Download the tar.gz file from http://sourceforge.net/projects/gnuplot/files/
- tar xvzf tar.gz file
- sudo python setup.py install

2.1.4 Installing gnuplot - Method 2

- sudo apt-get install gnuplot

2.2 For Fedora Users

2.2.1 Installing wxPython (as root user)

- yum install wxPython (not it is wxPython not wxpython)

2.2.2 Installing numpy (as root user)

- yum install numpy

2.2.3 Installing gnuplot (as root user)

- Download the tar.gz file from http://sourceforge.net/projects/gnuplot/files/
- tar xvzf tar.gz file
- python setup.py install

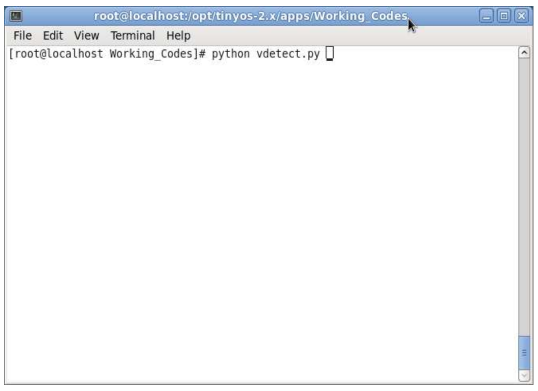
3 Installing VDetect Tool

In order to run this tool, copy the vdetect.py file and paste it in the /opt/tinyos-2.x/apps/ directory. Go to the command prompt and go to /opt/tinyos-2.x/apps/ directory. Type "python vdetect.py" (without the quotes) in the command line (as shown in Figure 1). This should pop out another window as shown in Figure 2. The code for the vdetect.py file is explained later in the Appendix part.

4 Features of VDetect Tool

In this section we shall look at the features this tool presents. The VDetect Tool is divided into four windows as shown in Figure 2.

- First Window - It displays the tree directory structure. The location where the vdetect.py file is place is the default location. It is advised to put the vdetect.py file in the /opt/tinyos-2.x/apps/ directory path. This makes the installation process simpler.
- Second Window - It is a text box and displays the text when a mote is loaded with code or displays the values of the magnetometer readings based on the users code.
- Third Window - It shows the contents or files present in a particular directory. If there are multiple directories present in a directory, it displays the list of directories present in the parent directory. If there are files in the parent directory, it displays the list of files in that directory.
- Fourth Window - This window has many functionality buttons. It can be used to load the mote (telosb or iwise) with the code. It can also be used to plot the data or save the data that is displayed in the Second Window. The plot function plots the real time magnetometer data.


root@localhost:/opt/tinyos-2.x/apps/Working_Codes
File Edit View Terminal Help

```
[root@localhost Working_Codes]# python vdetect.py
```

Figure 1: On the terminal, type "python vdetect.py" without the quotes to run the tool

5 Functions of VDetect Tool

The main objective of this tool is to make the data collection process easier. We have introduced several features in this tool in order to do that as shown in Figure 3. There are two basic steps to be followed in order to program the mote.

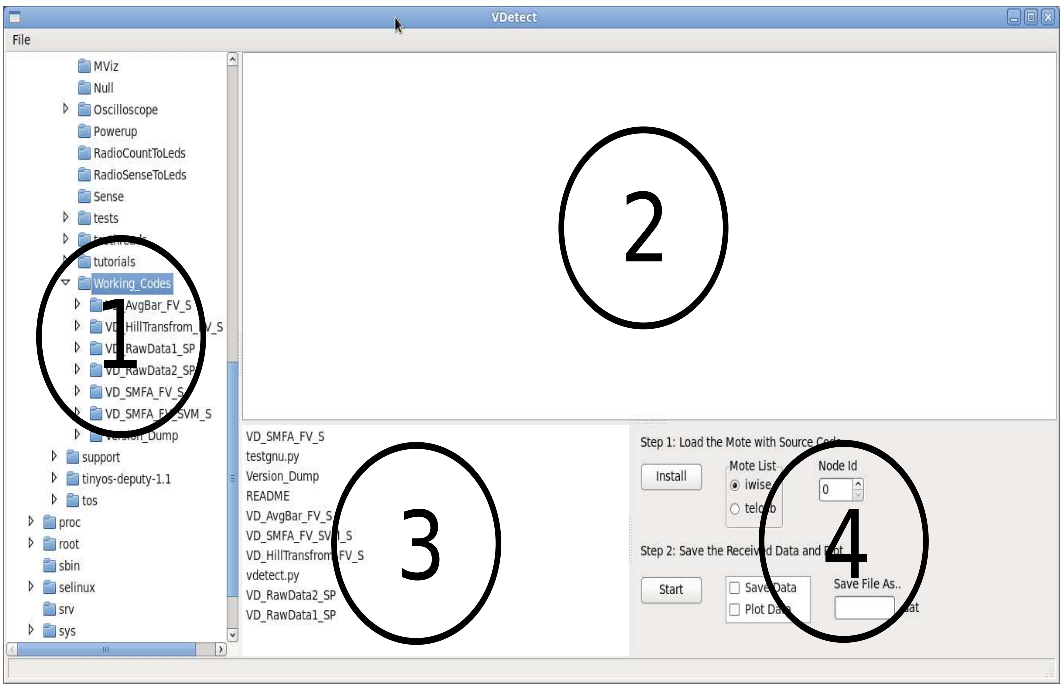


Figure 2: The VDetect Tool divided into four windows. First window displays the folder structure. Second window displays the text box. Third window displays the contents of the directory selected. Fourth window represents the functionality tools available in the tool

- Step 1: Load the mote with the source code or the application. This involves loading both the transmitter and the receiver mote. The process to be followed can be seen in the later sections.
- Step 2: Save the received data and/or Plot. This step mainly involves with saving and visualizing the data.

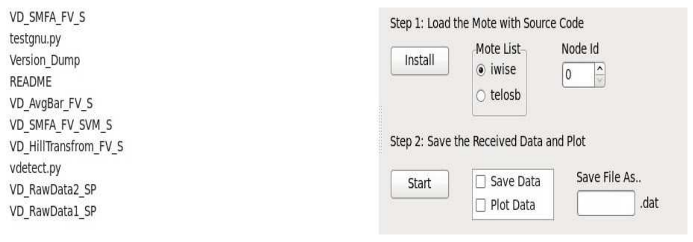


Figure 3: The different functionality of the VDetect Tool box

6 Steps To Be Followed

In all the codes we have followed a transmitter (tx) and receiver (rx) model. The CDAC iWiSe is used as a transmitter and a TelosB is used as a receiver mote. We have designed several versions of codes in order to test which feature extraction algorithm suits the best. The codes are available in the **vdetect_space** folder located in the **/opt/tinyos-2.x/apps** directory.

The general naming formate followed is **VD_<ApplicationName>_SP**, where **VD** stands for Vehicle Detection, **SP** stands for Store and Plot. Some of the applications may have only Store data functionality.

6.1 List of Directories and their Functionality

- **VD_RawData1_SP**: In this the tx (CDAC iWiSe mote) captures the magnetic readings using a HMC1502 magnetometer and transmits it to the rx (TelosB mote). The vehicle detection algorithm is coded in the rx mote. The output consists of the magnetic readings, state of the ATDA algorithm, packet count, OverThreshold Flag, number of metallic object detected. *Note*: While printing the magnetic readings it is necessary that they are printed as "MagY" and "MagZ". This is done so that the VDetect Tool can easily distinguish the magnetic readings from others and also enables the Plot option to visualize the real-time magnetic data.

- **VD_RawData2_SP**: In this the tx is coded with the ATDA vehicle detection algorithm. Only in the presence of a metallic object, the transmitter transmits the data to the rx. This is done so as to minimize the energy consumption of the system. The rx receives the data, which is the magnetic readings of the vehicle after applying the ATDA algorithm or the vehicle detection algorithm.

- **VD_mATDA_S**: In this the tx is coded with the m-ATDA vehicle detection algorithm as discussed in the earlier chapters. Here we make use of the readings of only one magnetometer axis instead of two. Only in the presence of a metallic object, the transmitter transmits the data to the rx. This is done so as to minimize the energy consumption of the system. The rx receives the data, which is the magnetic readings of the vehicle after applying the m-ATDA algorithm or the vehicle detection algorithm.

- **VD_AvgBar_FV_S**: In this the tx is coded not only with the vehicle detection algorithm but also with the feature extraction algorithm. We use Average Bar algorithm in this case. The size of the feature vector can be tuned. We have fixed it to 10 in our current application. The rx receives the number of vehicles detected and the feature vector. This can be a very useful application when it comes collecting data in order to train the motes.

- **VD_AvgBar_FV_SVM_S**: In this, apart from the functions mentioned above, this application also classifies a feature vector into either of the two categories - Type 1 (length of the car $\in$ [3.0,3.5]m) or Type 4 (length of the car $\in$ [>4.5]m). The receiver mote receives the feature vector, the vehicle count and the type of car the feature vector belongs to.

- **VD_HillTransform_FV_S**: In this the tx is coded not only with the vehicle detection algorithm but also with the feature extraction algorithm. We use Hill Transform algorithm in this case. The size of the feature vector can be tuned. We have fixed it to 10 in our current application. The rx receives the number of vehicles detected and the feature vector. This can be a very useful application when it comes collecting data in order to train the motes.

- **VD_HillTransform_FV_SVM_S**: In this, apart from the functions mentioned above, this application also classifies a feature vector into either of the two categories - Type 1 (length of the car $\in$ [3.0,3.5]m) or Type 4 (length of the car $\in$ [>4.5]m). The receiver mote receives the feature vector, the vehicle count and the type of car the feature vector belongs to.

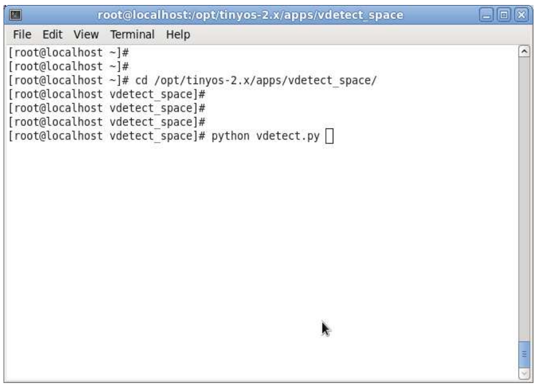
- **VD_SMFA_FV_S**: In this the tx is coded not only with the vehicle detection algorithm but also with the feature extraction algorithm. We use Segmented Magnetic Field Angle (SMFA) algorithm in this case. We have fixed the Length of the segment $L = 1$ and the size of the bin $Q = 5$. The rx receives the number of vehicles detected and the feature vector. This can be a very useful application when it comes collecting data in order to train the motes.

- **VD_SMFA_FV_SVM_S**: In this, apart from the functions mentioned above, this application also classifies a feature vector into either of the two categories - Type 1 (length of the car $\in$ [3.0,3.5]m) or Type 4 (length of the car $\in$ [>4.5]m). The receiver mote receives the feature vector, the vehicle count and the type of car the feature vector belongs to.

6.2 Example 1: VD_RawData1_SP

In this subsection, we will demonstrate the working of the tx / rx model using the VDetect Tool. We will also show how to load the motes and print data without using the VDetect Tool.

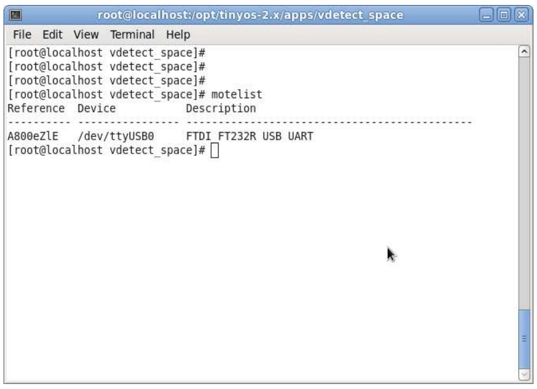
- **Step 1**: Navigate to the /opt/tinyos-2.x/apps/vdetect_space directory using terminal and locate the vdetect.py file. In the command line type "python vdetect.py"


root@localhost:/opt/tinyos-2.x/apps/vdetect_space
File Edit View Terminal Help

```
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# cd /opt/tinyos-2.x/apps/vdetect_space/
[root@localhost vdetect_space]#
[root@localhost vdetect_space]#
[root@localhost vdetect_space]#
[root@localhost vdetect_space]# python vdetect.py
```

Figure 4: In the terminal, type “python vdetect.py” without the quotes to run the tool

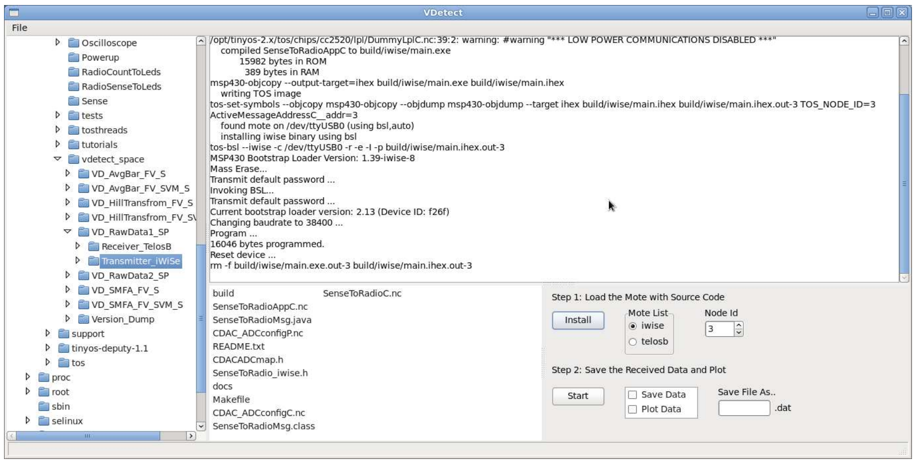
- **Step 2**: Open another terminal and type “motelist”. If it displays as “No devices were found” reconnect the mote to the USB port.


root@localhost:/opt/tinyos-2.x/apps/vdetect_space
File Edit View Terminal Help

```
[root@localhost vdetect_space]#
[root@localhost vdetect_space]#
[root@localhost vdetect_space]#
[root@localhost vdetect_space]# motelist
Reference  Device           Description
---------- ---------------- ---------------------------------------------
A800eZlE   /dev/ttyUSB0     FTDI FT232R USB UART
[root@localhost vdetect_space]#
```

Figure 5: In the terminal, type “motelist” without the quotes

- **Step 3**: Once the VDetect Tool loads, connect the CDAC iWiSe mote to any USB port of your computer machine. Make sure no other mote is connected to the USB port. In the Window 1 (as shown in Figure 2, you will be able to see the list of directories. Click on the arrow beside the **VD_RawData1_SP** to open the directory tree structure. In every application directory there is **Transmitter_iWiSe** directory and **Receiver_TelosB** directory. Since we have connected a CDAC iWiSe mote to the USB click on the **Transmitter_CDAC** directory as shown in the Figure 5. Now in the Window 4, select the mote. In this case it is iwise (which is the default option). In the Node Id, enter any number from 1 to 10 and click “Install”. Node Id 0 is used by the base station, which in this case is the rx mote. The command window - Window 2 shows the log messages. Make sure there are no errors in the log message.



```
/opt/tinyos-2.x/tos/chips/cc2520/lpl/DummyLplC.nc:39:2: warning: #warning "*** LOW POWER COMMUNICATIONS DISABLED ***"
   compiled SenseToRadioAppC to build/iwise/main.exe
         15982 bytes in ROM
          389 bytes in RAM
msp430-objcopy --output-target=ihex build/iwise/main.exe build/iwise/main.ihex
   writing TOS image
tos-set-symbols --objcopy msp430-objcopy --objdump msp430-objdump --target ihex build/iwise/main.ihex build/iwise/main.ihex.out-3 TOS_NODE_ID=3
ActiveMessageAddressC__addr=3
   found mote on /dev/ttyUSB0 (using bsl,auto)
   installing iwise binary using bsl
tos-bsl --iwise -c /dev/ttyUSB0 -r -e -I -p build/iwise/main.ihex.out-3
MSP430 Bootstrap Loader Version: 1.39-iwise-8
Mass Erase...
Transmit default password ...
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 2.13 (Device ID: f26f)
Changing baudrate to 38400 ...
Program ...
16046 bytes programmed.
Reset device ...
rm -f build/iwise/main.exe.out-3 build/iwise/main.ihex.out-3
```

Step 1: Load the Mote with Source Code

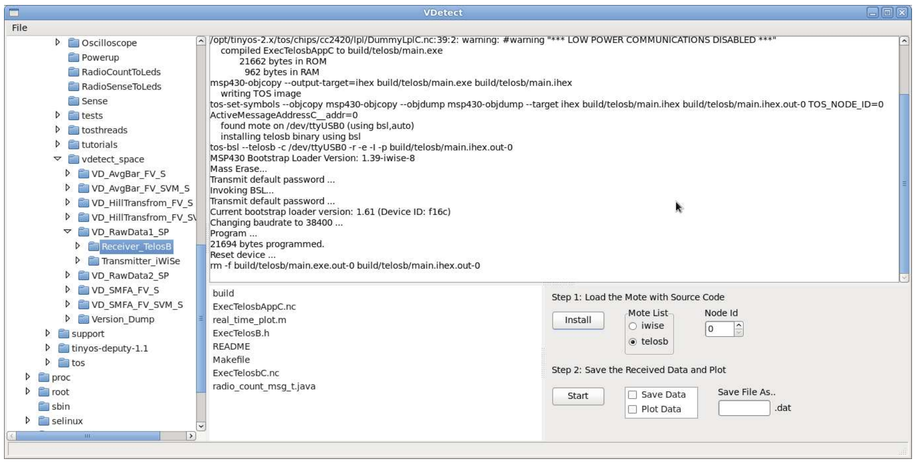
Mote List: [✓ iwise] telosb

Node Id: 3

Step 2: Save the Received Data and Plot

Figure 6: Loading the CDAC iWiSe mote.

- **Step 4**: Unplug the CDAC mote and connect the TelosB mote to the USB port. Repeat Step 2. Now in the VDetect Tool, click on the **Receiver_TelosB** folder in the parent **VD_RawData1_SP** directory. Change the Node Id to 0 and in the motelist select telosb. Now click "Install". The command window - Window 2 shows the log messages. Make sure there are no errors in the log message.



```
/opt/tinyos-2.x/tos/chips/cc2420/lpl/DummyLplC.nc:39:2: warning: #warning "*** LOW POWER COMMUNICATIONS DISABLED ***"
   compiled ExecTelosbAppC to build/telosb/main.exe
         21662 bytes in ROM
          962 bytes in RAM
msp430-objcopy --output-target=ihex build/telosb/main.exe build/telosb/main.ihex
   writing TOS image
tos-set-symbols --objcopy msp430-objcopy --objdump msp430-objdump --target ihex build/telosb/main.ihex build/telosb/main.ihex.out-0 TOS_NODE_ID=0
ActiveMessageAddressC__addr=0
   found mote on /dev/ttyUSB0 (using bsl,auto)
   installing telosb binary using bsl
tos-bsl --telosb -c /dev/ttyUSB0 -r -e -I -p build/telosb/main.ihex.out-0
MSP430 Bootstrap Loader Version: 1.39-iwise-8
Mass Erase...
Transmit default password ...
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 1.61 (Device ID: f16c)
Changing baudrate to 38400 ...
Program ...
21694 bytes programmed.
Reset device ...
rm -f build/telosb/main.exe.out-0 build/telosb/main.ihex.out-0
```

Step 1: Load the Mote with Source Code

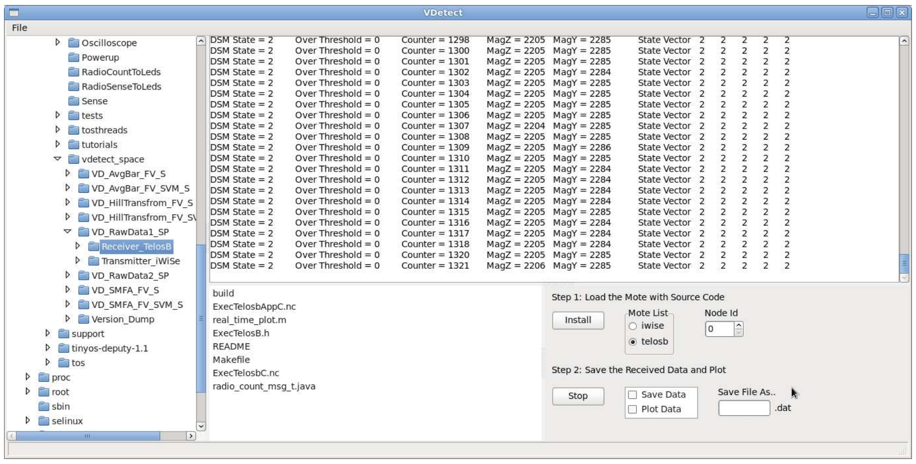
Mote List: iwise [✓ telosb]

Node Id: 0

Step 2: Save the Received Data and Plot

Figure 7: Loading the TelosB mote.

- **Step 5**: Now both the rx and the tx are ready. Deploy the transmitter such that it is within the range of the receiver. Make sure the transmitter battery is charged fully. Now click "Start" button. You will be able to see different parameters being displayed on Window 2.



```
DSM State = 2    Over Threshold = 0    Counter = 1298    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1300    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1301    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1302    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1303    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1304    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1305    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1306    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1307    MagZ = 2204   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1308    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1309    MagZ = 2205   MagY = 2286      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1310    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1311    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1312    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1313    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1314    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1315    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1316    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1317    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1318    MagZ = 2205   MagY = 2284      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1320    MagZ = 2205   MagY = 2285      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 1321    MagZ = 2206   MagY = 2285      State Vector  2    2    2    2    2
```

Step 1: Load the Mote with Source Code

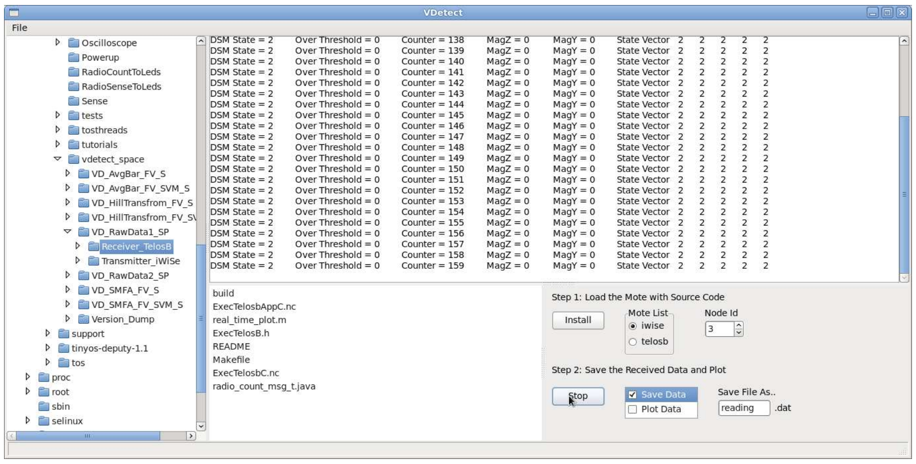
Mote List: [ ] iwise [✓ telosb] Node Id: 0

Step 2: Save the Received Data and Plot

[ ] Save Data [ ] Plot Data

Figure 8: On clicking "Start".

- **Step 6**: If you want to store the data that is being displayed on Window 2, then check the Save Data and give the file name. The file will be stored in .dat format in the **vdetect_space** directory.



```
DSM State = 2    Over Threshold = 0    Counter = 138    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 139    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 140    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 141    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 142    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 143    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 144    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 145    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 146    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 147    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 148    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 149    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 150    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 151    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 152    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 153    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 154    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 155    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 156    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 157    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 158    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
DSM State = 2    Over Threshold = 0    Counter = 159    MagZ = 0      MagY = 0      State Vector  2    2    2    2    2
```

Step 1: Load the Mote with Source Code

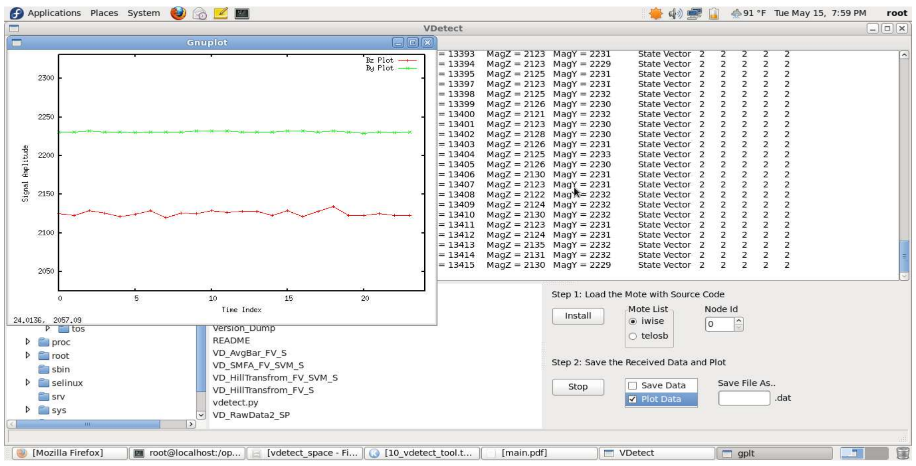
Mote List: [✓ iwise] [ ] telosb Node Id: 3

Step 2: Save the Received Data and Plot

[✓ Save Data] [ ] Plot Data Save File As.. reading .dat

Figure 9: Save Data.

- **Step 7**: If you want to plot the data (magnetometer readings) that is being displayed on Window 2, then check the Plot Data.



```
= 13393   MagZ = 2123   MagY = 2231      State Vector  2    2    2    2    2
= 13394   MagZ = 2123   MagY = 2229      State Vector  2    2    2    2    2
= 13395   MagZ = 2125   MagY = 2231      State Vector  2    2    2    2    2
= 13397   MagZ = 2123   MagY = 2231      State Vector  2    2    2    2    2
= 13398   MagZ = 2125   MagY = 2232      State Vector  2    2    2    2    2
= 13399   MagZ = 2126   MagY = 2230      State Vector  2    2    2    2    2
= 13400   MagZ = 2121   MagY = 2232      State Vector  2    2    2    2    2
= 13401   MagZ = 2123   MagY = 2230      State Vector  2    2    2    2    2
= 13402   MagZ = 2128   MagY = 2230      State Vector  2    2    2    2    2
= 13403   MagZ = 2126   MagY = 2231      State Vector  2    2    2    2    2
= 13404   MagZ = 2125   MagY = 2233      State Vector  2    2    2    2    2
= 13405   MagZ = 2126   MagY = 2230      State Vector  2    2    2    2    2
= 13406   MagZ = 2130   MagY = 2231      State Vector  2    2    2    2    2
= 13407   MagZ = 2123   MagY = 2231      State Vector  2    2    2    2    2
= 13408   MagZ = 2122   MagY = 2232      State Vector  2    2    2    2    2
= 13409   MagZ = 2124   MagY = 2232      State Vector  2    2    2    2    2
= 13410   MagZ = 2130   MagY = 2232      State Vector  2    2    2    2    2
= 13411   MagZ = 2123   MagY = 2231      State Vector  2    2    2    2    2
= 13412   MagZ = 2124   MagY = 2231      State Vector  2    2    2    2    2
= 13413   MagZ = 2135   MagY = 2232      State Vector  2    2    2    2    2
= 13414   MagZ = 2131   MagY = 2232      State Vector  2    2    2    2    2
= 13415   MagZ = 2130   MagY = 2229      State Vector  2    2    2    2    2
```

Step 1: Load the Mote with Source Code

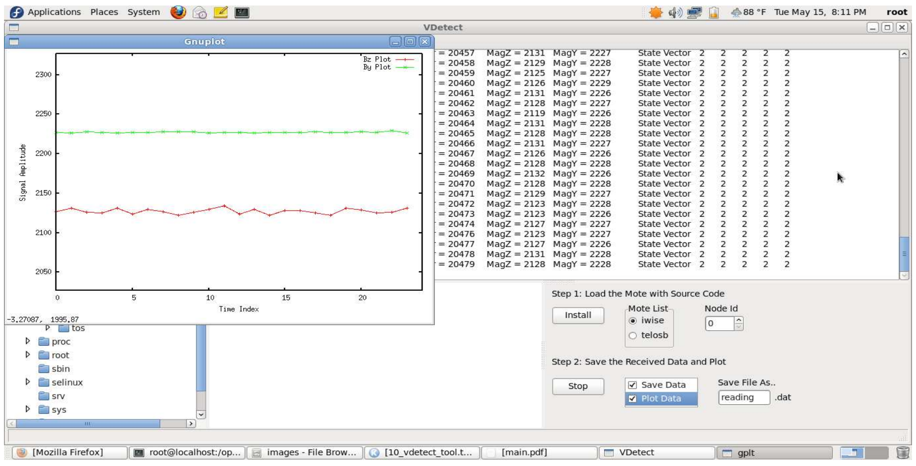
Mote List: [✓ iwise] [ ] telosb Node Id: 0

Step 2: Save the Received Data and Plot

[ ] Save Data [✓ Plot Data]

Figure 10: Plot Data.

- **Step 8**: One can also plot and save the data at once. In order to do that check both "Plot Data" and "Save Data" box. File name is to be mentioned in the space provided. Click on the "Start" button.



```
= 20457   MagZ = 2131   MagY = 2227      State Vector  2    2    2    2    2
= 20458   MagZ = 2129   MagY = 2228      State Vector  2    2    2    2    2
= 20459   MagZ = 2125   MagY = 2227      State Vector  2    2    2    2    2
= 20460   MagZ = 2126   MagY = 2229      State Vector  2    2    2    2    2
= 20461   MagZ = 2131   MagY = 2226      State Vector  2    2    2    2    2
= 20462   MagZ = 2128   MagY = 2227      State Vector  2    2    2    2    2
= 20463   MagZ = 2119   MagY = 2226      State Vector  2    2    2    2    2
= 20464   MagZ = 2131   MagY = 2228      State Vector  2    2    2    2    2
= 20465   MagZ = 2128   MagY = 2228      State Vector  2    2    2    2    2
= 20466   MagZ = 2131   MagY = 2227      State Vector  2    2    2    2    2
= 20467   MagZ = 2126   MagY = 2226      State Vector  2    2    2    2    2
= 20468   MagZ = 2128   MagY = 2228      State Vector  2    2    2    2    2
= 20469   MagZ = 2132   MagY = 2226      State Vector  2    2    2    2    2
= 20470   MagZ = 2128   MagY = 2228      State Vector  2    2    2    2    2
= 20471   MagZ = 2129   MagY = 2227      State Vector  2    2    2    2    2
= 20472   MagZ = 2123   MagY = 2228      State Vector  2    2    2    2    2
= 20473   MagZ = 2123   MagY = 2226      State Vector  2    2    2    2    2
= 20474   MagZ = 2127   MagY = 2227      State Vector  2    2    2    2    2
= 20476   MagZ = 2123   MagY = 2227      State Vector  2    2    2    2    2
= 20477   MagZ = 2127   MagY = 2226      State Vector  2    2    2    2    2
= 20478   MagZ = 2131   MagY = 2228      State Vector  2    2    2    2    2
= 20479   MagZ = 2128   MagY = 2228      State Vector  2    2    2    2    2
```

Step 1: Load the Mote with Source Code

Install

Mote List: [✓ iwise] telosb

Node Id: 0

Step 2: Save the Received Data and Plot

Stop

[✓ Save Data]
[✓ Plot Data]

Save File As..: reading .dat

Figure 11: Save and Plot Data.

7 Implementation Code

```
#!/usr/bin/python
#!/bin/bash
#!/usr/local/bin/gnuplot -persist

################################################################################
# Import Packages
################################################################################
import os
import wx
import string
import commands
import thread
import time
import threading
import subprocess
import sys

try:
    import Gnuplot, Gnuplot.PlotItems, Gnuplot.funcutils
except ImportError:
    # kludge in case Gnuplot hasn’t been installed as a module yet:
    import __init__
    Gnuplot = __init__
    #import PlotItems
    #Gnuplot.PlotItems = PlotItems
    #import funcutils
    #Gnuplot.funcutils = funcutils

################################################################################
# Function
################################################################################

class DragDrop(wx.Frame):

    # OnSelect of the directory
    def OnSelect(self, event):
        list = os.listdir(self.dir.GetPath())
        self.lc2.ClearAll()
        for i in range(len(list)):
            if list[i][0] != ’.’:
                self.lc2.InsertStringItem(0, list[i])

    # OnAbout Button
    def OnAbout(self,event):
        # Create a message dialog box
        dlg = wx.MessageDialog(self, " A Vehicle Detector \n and Classifier...
        Interface\n By Prateek and K V S Hari", "VDetect", wx.OK)
        dlg.ShowModal() # Shows it
        dlg.Destroy() # finally destroy it when finished.

    # OnExit Button
    def OnExit(self,event):
        self.Close(True)  # Close the frame.

    # OnOpen Button
    def OnOpen(self,event):
        """ Open a file"""
    dlg = wx.FileDialog(self, "Choose a file", self.dirname, "", "*.*", wx.OPEN)
    if dlg.ShowModal() == wx.ID_OK:
        self.filename = dlg.GetFilename()
        self.dirname = dlg.GetDirectory()
        f = open(os.path.join(self.dirname, self.filename), 'r')
        self.lc1.SetValue(f.read())
        f.close()
    dlg.Destroy()

# OnMoteButton - When Install is clicked
def OnMoteButton(self,event):

    self.home = os.getcwd()
    mote = self.sbs.GetStringSelection()
    nodeid = self.spinner.GetValue()
    os.chdir(self.dir.GetPath())
    command = 'make ' + mote + ' install,' + str(nodeid)
    self.display_text = commands.getstatusoutput(command)
    self.display_text = self.display_text[1].split('\n')
    self.lc1.Clear()

    self.NewThread = threading.Thread(target = self.LongRunning)
    self.NewThread.start()
    os.chdir(self.home)

def LongRunning(self):

    Counter = 0
    while True:
            time.sleep(0.01)
            if len(self.display_text) == Counter:
                    break
            wx.CallAfter(self.UpdateMultiLine, self.display_text[Counter] + "\n")
            Counter = Counter + 1

def UpdateMultiLine(self, data):

    self.lc1.AppendText(data)
    self.lc1.SaveFile("plotdata", fileType=wx.TEXT_TYPE_ANY)

def PrintText(self):

    fp1 = os.open('temp',os.O_RDWR)
    fq = os.fdopen(fp1, "r")
    time.sleep(1)
    line = fq.readlines(2)
    while self.child_poll == None:
            time.sleep(0.005)
            line = fq.readline()
            wx.CallAfter(self.UpdateMultiLine, line)
            if self.btnLabel == "Stop":
                    break
    fq.close()
    os.system('rm -f temp')

def PlotData(self):

    time.sleep(2)
    fp2 = os.open('plotdata',os.O_RDWR)
    fr = os.fdopen(fp2, "r")
    line1 = fr.readline(2)
    N = 25
    Counter = 0
    Bz = []
    By = []

    g = Gnuplot.Gnuplot()
    g.clear()

    while True:
            Counter = Counter + 1;
            time.sleep(0.1)

            if len(line1) != 0 and (line1.find('MagZ') != -1 and line1.find('MagY') != -1):

                    line1 = line1.split()
                    Mz = int(line1[line1.index('MagZ')+2])
                    My = int(line1[line1.index('MagY')+2])
                    if Counter <= N:
                            Bz.append(Mz);
                            By.append(My);
                            line1 = fr.readline()
                    else:
                            mean_bz = sum(Bz)/len(Bz)
                            mean_by = sum(By)/len(By)
                            g.set_range('xrange',(0,N-1))
                            g.set_range('yrange',(min(mean_bz,mean_by)-100,...
                            max(mean_bz,mean_by)+100))
                            p1 = Gnuplot.Data((Bz), title='Bz Plot',...
                            with_='linesp', inline=1)
                            p2 = Gnuplot.Data((By), title='By Plot',...
                            with_='linesp', inline=1)
                            g.xlabel('Time Index')
                            g.ylabel('Signal Amplitude')
                            g.plot(p1,p2)
                            del Bz[0];
                            del By[0]
                            Bz.append(Mz)
                            By.append(My)
                            line1 = fr.readline()
            else:
                     line1 = fr.readline()

            if not line1:
                    time.sleep(0.001)
                    continue

            if self.btnLabel == "Stop":
                    g.close()
                    os.system('rm -f plotdata')
                    break
    fr.close()


def OnToggleButton(self, event):

    self.checked_string = self.checklist_box.GetCheckedStrings()
    self.btnLabel = self.toggleBtn.GetLabel()
```

```
        if self.btnLabel == "Start":

                self.toggleBtn.SetLabel("Stop")
                self.fp = open("temp", "w")
                child = subprocess.Popen(["java", "net.tinyos.tools.PrintfClient",...
                "-comm", "serial@/dev/ttyUSB0:telosb"], shell=False, stdout=self.fp)
                self.child_pid = child.pid
                self.child_poll = child.poll()
                self.lc1.Clear()

                if (len(self.checked_string) == 1 or len(self.checked_string) == 2)...
                and ( "Plot Data" in self.checked_string ):
                        NewThread2 = threading.Thread(target = self.PlotData)
                        NewThread2.start()

                NewThread1 = threading.Thread(target = self.PrintText)
                NewThread1.start()


        if self.btnLabel == "Stop":
                os.system('kill -9 ' + str(self.child_pid))
                self.fp.close()

                if (len(self.checked_string) == 1 or len(self.checked_string) == 2)...
                and self.checked_string[0] == 'Save Data':
                        fname = self.file_name.GetValue()
                        fname = fname + '.dat'
                        os.system('cp temp ' + fname)


                os.system('rm -f temp')
                os.system('rm -f plotdata')
                self.toggleBtn.SetLabel("Start")

################################################################################
# Main Panel
################################################################################

    def __init__(self, parent, id, title):

        wx.Frame.__init__(self, parent, id, title, pos=(0, 0), size=wx.DisplaySize())
        self.dirname=''

        # Creating status bar
        self.CreateStatusBar() # A Statusbar in the bottom of the window

        filemenu= wx.Menu()
        menuOpen = filemenu.Append(wx.ID_OPEN, "&Open"," Open a file to edit")
        menuAbout= filemenu.Append(wx.ID_ABOUT, "&About"," Information about program")
        menuExit = filemenu.Append(wx.ID_EXIT,"E&xit"," Terminate the program")

        # Creating the menubar.
        menuBar = wx.MenuBar()
        menuBar.Append(filemenu,"&File") # Adding the "filemenu" to the MenuBar
        self.SetMenuBar(menuBar)  # Adding the MenuBar to the Frame content.

        self.Bind(wx.EVT_MENU, self.OnOpen, menuOpen)
        self.Bind(wx.EVT_MENU, self.OnExit, menuExit)
        self.Bind(wx.EVT_MENU, self.OnAbout, menuAbout)

        # SplitterWindow is used to split into different windows
        splitter1 = wx.SplitterWindow(self, -1, style=wx.SP_3D)
        splitter2 = wx.SplitterWindow(splitter1, -1, style=wx.SP_3D)
        splitter3 = wx.SplitterWindow(splitter2, -1, style=wx.SP_3D)

        # self.dir is the directory list
        # self.lc1 is the multiline command window
        # self.lc2 is the list of files in a directory
        # self.lc3 is the fixed panel with buttons
        self.dir = wx.GenericDirCtrl(splitter1, -1, dir=os.getcwd(),...
        style=wx.DIRCTRL_DIR_ONLY)
        self.lc2 = wx.ListCtrl(splitter3, -1, style=wx.LC_LIST,...
        size = wx.Size(500,-1))
        self.lc1 = wx.TextCtrl(splitter2, -1,...
        style=wx.TE_MULTILINE|wx.TE_READONLY|wx.TE_AUTO_URL)
        self.lc3 = wx.Panel(splitter3, -1)

        tree = self.dir.GetTreeCtrl()

        splitter3.SplitVertically(self.lc2, self.lc3)
        splitter2.SplitHorizontally(self.lc1, splitter3, -250)
        splitter1.SplitVertically(self.dir, splitter2, 300)

        self.Bind(wx.EVT_TREE_SEL_CHANGED, self.OnSelect, id=tree.GetId())

        # Panel and its buttons with functions for loading the motes
        wx.StaticText(self.lc3, -1, label="Step 1: Load the Mote with...
        Source Code", pos=(10,10))
        mote_button = wx.Button(self.lc3, label="Install", pos=(10,40), size=(80,30))
        self.sbs = wx.RadioBox(self.lc3, -1, label='Mote List', pos=(120,35), ...
        size=(75,75), choices=['iwise','telosb'], style=wx.RA_SPECIFY_ROWS)
        wx.StaticText(self.lc3, -1, label='Node Id', pos=(240,35), style=wx.ALIGN_CENTER)
        self.spinner = wx.SpinCtrl(self.lc3, -1, "", pos=(240,55), size=(60,-1))
        self.spinner.SetRange(0,10)
        self.spinner.SetValue(0)
        self.Bind(wx.EVT_BUTTON, self.OnMoteButton, mote_button)

        # Panel and its buttons with functions for plotting and recording the data
        wx.StaticText(self.lc3,-1, label="Step 2: Save the Received Data and Plot",...
        pos=(10,125))
        self.toggleBtn = wx.Button(self.lc3, wx.ID_ANY, label="Start", pos=(10,160),...
        size=(80,30))
        self.checklist_box = wx.CheckListBox(self.lc3, -1, pos=(120,160),...
        size=(110,50), choices=['Save Data', 'Plot Data'])
        self.toggleBtn.Bind(wx.EVT_BUTTON, self.OnToggleButton)
        wx.StaticText(self.lc3, -1, label="Save File As..", pos=(260,160))
        self.file_name = wx.TextCtrl(self.lc3, wx.ID_ANY, pos=(260,180))
        wx.StaticText(self.lc3, -1, label=".dat", pos=(345,185))

        self.OnSelect(0)
        self.Centre()
        self.Show(True)

app = wx.App()
DragDrop(None, -1, 'VDetect')
app.MainLoop()
```

8 Acknowledgements

This project is funded by DIT-ASTEC Wireless Sensor Project, Department of Information Technology, Ministry of Communications & Information Technology, Govt. of India.