

**VEHICLE DETECTION AND CLASSIFICATION USING
MAGNETOMETER – Data Acquisition**

Team: Akshath S Bhat, Arvind K Deshpande, Kaustubh G Deshpande

Guide: Prof. K V S Hari

Statistical Signal Processing Lab

Department of Electrical Communication Engineering

Indian Institute of Science

Contents

1 Introduction	1
1.1 Basic idea	1
1.2 Objective	1
2 Anisotropic Magnetoresistive Sensor	2
2.1 Theory of the AMR Sensor	2
3 Hardware Description	6
3.1 SBT80 Sensor Board	6
3.2 TELOSB mote Platform	7
3.2.1 Applications	8
4 Software Description	10
5 Vehicle Detection	11
5.1 Magnetic Signal Analysis	11
5.2 Data Collection	11
5.3 Modified Setup Similar to Actual Road Scenario	12
5.3.1 Experimentation using Remote Controlled Car	12
5.3.2 Experimentation using a Skate Board	15
5.3.2.1 Experiment no. 1	17
5.3.2.2 Experiment no. 2	19
5.4 Organising the data collected into Files/Directories	26
Appendix A	
Appendix B	

List of Figures

2.1 AMR Strip	3
2.2 Wheatstone Bridge Construction	5
3.1 SBT80 Sensor Board	6
3.2 TelosB mote attached with SBT80 Sensor Board	7
5.1 Distortion in earth's magnetic field caused by a vehicle	11
5.2 Magnetometer setup in actual road scenario	12
5.3.1 Top view of Remote Controlled Car	13
5.3.2 Side View of Remote Controlled Car	14
5.3.3 Three paths in which setup is passed	14
5.3.4 Top view of setup mounted on the Skate Board	16
5.3.5 Side view of setup mounted on the Skate Board	16
5.3.6 Waveforms obtained without fiber box	17
5.3.7 Waveforms obtained with fiber box	18
5.3.8 Waveforms obtained when mote is mounted on Skate Board	19
5.3.9 Waveforms obtained when mote is fixed to bottom face of	20
Skate Board	
5.3.10 Top view of setup fixed to the bottom face of the Skate Board ...	21
5.3.11 Side view of the setup fixed to the bottom face of the Skate Board ..	21

List of Tables

5.3 All Cars covered by using the two setups	22
--	----

VEHICLE DETECTION AND CLASSIFICATION USING MAGNETEOMETER

1. Introduction

The network of roads of a nation determine how well the entire nation is connected. It is an index of the rate of development of the country. Any developed or developing nation like India has a good network of roads. It is also true that there are traffic congestions especially cities and towns. The number of man hours wasted due to traffic congestion and the amount of pollution are significantly huge. Therefore, there is a great demand for Intelligent Traffic Systems which are capable of monitoring road traffic to reduce delay and to smoothen the flow of vehicles.

There are several different possibilities for a road traffic monitoring system. Induction loops, infrared sensing, pressure sensors, video cameras, to name a few, are technologies that can be used. However, some of the important characteristics which need to be taken into consideration for large scale deployment are cost, ease of deployment, flexibility in deployment, maintenance, life-time of the sensors used and power consumption. Wireless sensor network technology fits these requirements almost perfectly.

1.1 Basic idea

At any given place on the earth, there is a natural magnetic field due to earth having a suitable magnitude and direction. There is a fluctuation in the magnitude, but for practical cases it is assumed to be constant considering the degree of variation.

Whenever a ferromagnetic substance enters the vicinity of the earth's natural magnetic field, there is a distortion in the pre-existing field. The commonly used vehicles on road are all having a definite metallic (ferromagnetic) content, may it be the engine or the chassis. Thus, when a vehicle enters the vicinity of a road, it causes distortions in the pre-existing field due to earth.

The size and shape, thus metallic content of each and every car is unique to the particular model of a brand/maker of the car. Thus the magnetic field distortions associated with each car is unique. If there is a way to record these fluctuations, then a signal can be associated to each vehicle and the signal can be termed as signature signal.

1.2 Objective:

The signature signal is a characteristic of a car and can be used to detect and also classify the cars, if there is a knowledge of the signature signals associated with the commonly used vehicles. The detection and classification of the vehicles can add intelligence to the current day traffic and can solve most of the problems.

2. Anisotropic Magnetoresistive Sensor

Magnetic sensors have typically been used for direction finding or navigation, but other applications have evolved over time. The technology for sensing magnetic fields has also evolved driven by the need for improved performance. One of the newer types of silicon based magnetic sensors are the Anisotropic Magnetoresistive (AMR)sensors. AMR sensors detect the distortions of the Earth's magnetic field, which is uniform over a wide area on the scale of kilometres.

A wide scale deployment of a wireless network of anisotropic magnetic sensors is envisioned. These passive sensors are mounted on low power consuming wireless transceivers called motes capable of communicating with each other and a basestation. The sensors are capable of sensing the magnetic field. Therefore, the field induced due to a large permeable object, like a vehicle, can be sensed. The idea is to place an array of these sensors in each lane, near the traffic lights and at a certain buffer length away from the traffic lights. The sensors are capable of detecting vehicles and a record of the number of vehicles going past it can be recorded. Thus, a rough estimate of the no. of vehicles in the buffer can be made. This estimate of the queue length can be sent to the traffic light controller, via the network of wireless sensors. The traffic controller uses the information from all the lanes and adaptively controls the traffic light configuration to reduce average delay and increase throughput. One can also think of a more centralized control system which manages the configuration of several successive traffic junctions.

2.1 Theory of the AMR Sensor

An AMR sensor is made by depositing a very thin layer of Permalloy on a Silicon wafer, patterned as a resistive strip. The properties of the thin AMR film result in a change in the resistance when an external field is applied.

The magnetization M in a ferromagnetic material always has the magnitude of the saturation magnetization M_S and only changes in direction. The theory of the AMR effect can, therefore, be split into two parts

- Electrical resistivity as a function of the direction of magnetization (θ_M) in the resistive strip
- The relation between direction of magnetization and externally applied field (H_{ext})

Consider a magnetoresistive strip as shown in Figure 2.1. The electrical resistance R is given by,

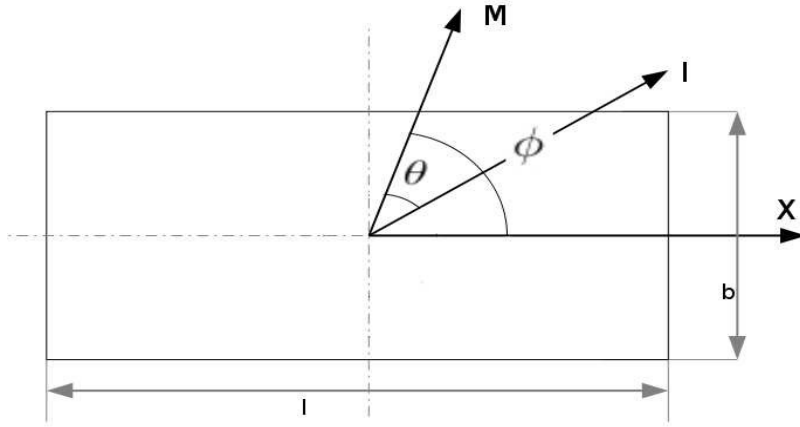


Figure 2.1: AMR strip

$$R = \rho_{0,n} \frac{l}{bd} + \Delta\rho \frac{l}{bd} \cos^2\theta = R_{0,n} + \Delta R \cos^2\theta \quad (2.1)$$

where $\rho_{0,n}$ and $\Delta\rho$ are material constants and θ is the angle between the current density vector and the magnetization vector. $R_{0,n}$ is the resistance when the current is perpendicular to the magnetization and ΔR is the maximum change in the resistance due to the magnetic field.

The spontaneous magnetization lies in the easy axis direction which is fixed by shape anisotropy. A magnetic field along the heavy axis (perpendicular to the easy axis) provides a rotation of the magnetization in the permalloy strip and thus a change of its resistance. Let us assume that the easy axis is along the X-axis. With the total anisotropy field $H_0 = \frac{2K}{\mu_0 M_s}$, the angle ϕ between \vec{M} and the easy axis is:

$$\sin \phi = \frac{H_y}{H_0} \quad (2.2)$$

for $-1 < \frac{H_y}{H_0} < 1$, where H_y is the Y-component of the applied field. Outside this range,

$$\sin \phi = \text{sign}\left(\frac{H_y}{H_0}\right)$$

We introduce two new resistances $R_{0,p}$ and R_0 :

$$R_{0,p} = R_{0,n} - \Delta R \quad (2.3)$$

$$R_0 = \frac{R_{0,p} + R_{0,n}}{2} \quad (2.4)$$

R_0 is the average resistance. Using Equations (2.1) and (2.3),

$$R(\theta) = R_{0,p} - \Delta R \sin^2 \theta \quad (2.5)$$

Using Equation (2.2), when the current flow is along the easy axis, we have

$$R(\theta) = R_{0,p} - \Delta R \left(\frac{H_y}{H_0} \right)^2 \quad (2.6)$$

for $|H_y| \leq H_0$ and

$$R(H_y) = R_{0,p} \quad (2.7)$$

for $|H_y| > H_0$. The resistance exhibits a strongly non-linear dependence on the external field. Also, the sign of H_y cannot be determined since R is a function of H_y . In order to alleviate these disadvantages, barber pole structures[1] are introduced. Mathematically, barber poles are represented by introducing an additional angle $\psi = 45^\circ$, which represents the angle between the easy axis and the current.

The angle θ in this case is

$$\theta = \phi - \psi \quad (2.8)$$

Therefore, the characteristics of a barber-pole AMR sensor are

$$R(H_y) = R_0 + \Delta R \frac{H_y}{H_0} \cdot \sqrt{1 - \left(\frac{H_y}{H_0} \right)^2} \quad (2.9)$$

For $|H_y| < \frac{H_0}{2}$, it is fairly linear with a non-linearity of less than 5%. In most applications, a single AMR-Resistor is not suited because it does not provide a zero reference and has a temperature coefficient of +0.3%/K.

The sensor is realized as a Wheatstone bridge with four individual resistors, as shown in Figure 2.2. The resistance changes are converted into a voltage output without a dc component.

This setup also compensates for the temperature dependence of the resistors. The output voltage of a Wheatstone bridge can be described by:

$$V_{out} = V_{in} \Delta R \frac{Hy}{H_0} \text{ sq. rt}(1 - (\frac{Hy}{H_0})^2) \quad (2.10)$$

One such AMR bridge can be used to measure the field along one direction. If a magnetometer comprises of two such AMR bridges placed perpendicular to each other, it can be used to measure the field along 2 orthogonal axes; similarly a 3-axis sensor.

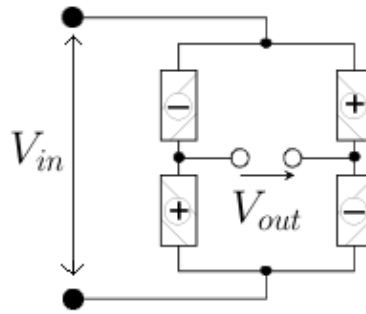
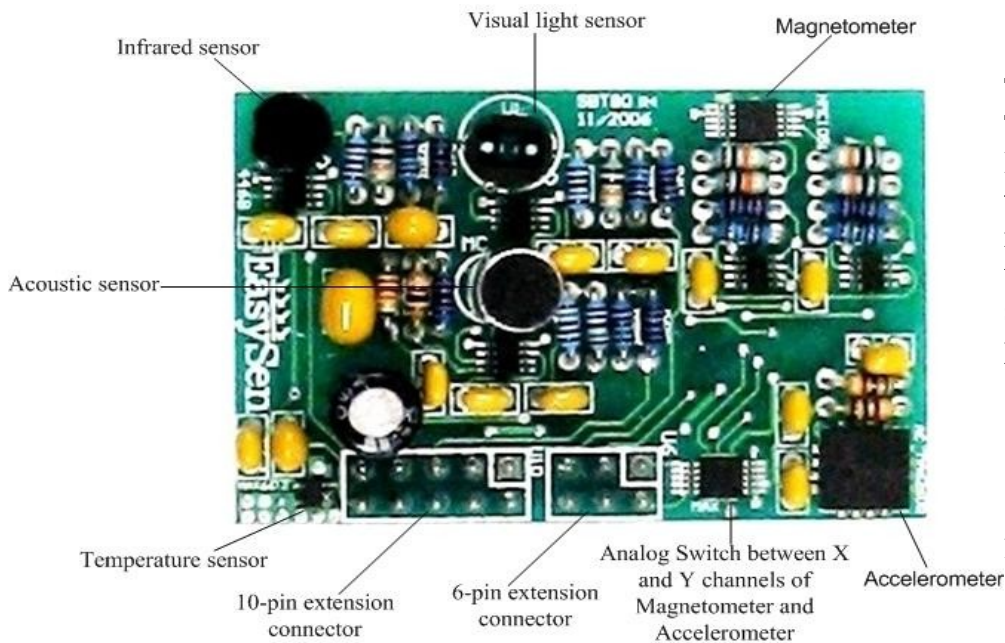


Figure 2.2: Wheatstone bridge construction.

3. Hardware Description

3.1 SBT80 Sensor Board



SBT80 Multi-Modality Sensor Board is used for TelosB wireless motes. This sensor board has HMC 1052 magnetometer which is a dual axis magnetometer. In addition to this it also has the other sensors used for different purposes.

Figure 3.1: SBT80 Sensor Board

The features of the SBT80 Sensor Board are:

- Sensor board for TelosB wireless sensor platform
 - Can be directly plugged in to the external connector of TelosB motes
- Eight Sensor channels with high sensitivity sensors
 - visual light, infrared, acoustic, temperature, dual-axis magnetometer and dual-axis accelerometer
- Smaller form factor than TelosB motes
- Power saving mode
- TinyOS and Java code support available
- Can be used for multi-modal surveillance and monitoring applications using sensor networks

3.2 TELOSB Mote Platform

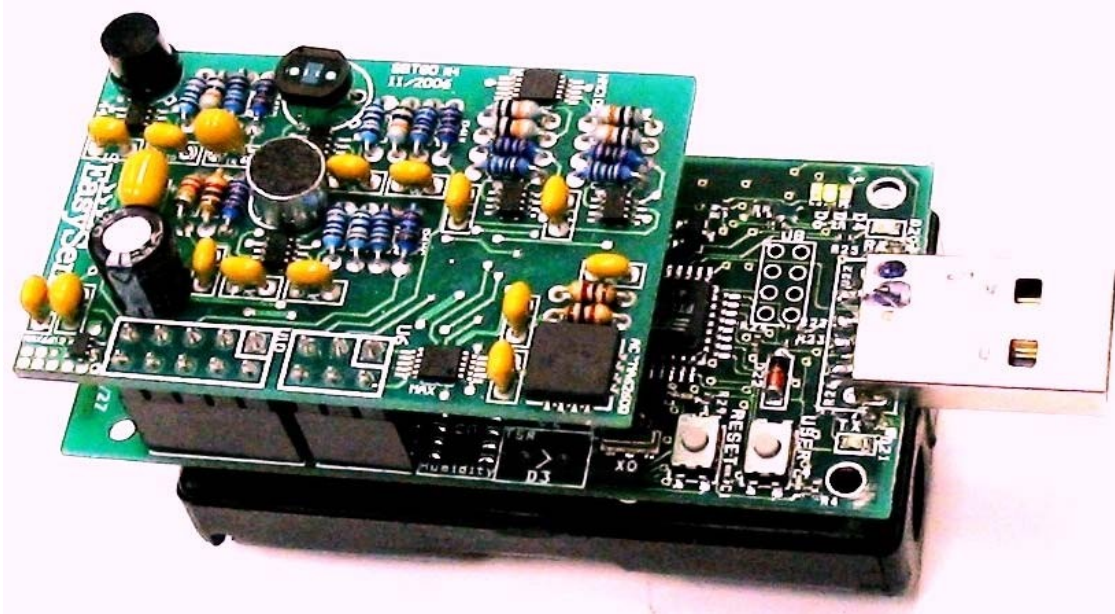


Figure 3.2: TelosB mote attached with SBT80 Sensor board

TELOSB MOTE PLATFORM FEATURES

- IEEE 802.15.4 Compliant
- 250 kbps, High Data Rate Radio
- TI MSP430 Microcontroller with 10kB RAM
- Integrated Onboard Antenna
- Data Collection and Programming via USB Interface
- Open-source Operating System
- Integrated Temperature, Light and Humidity Sensor

3.2.1 Applications

- Platform for Low Power Research Development
- Wireless Sensor Network Experimentation

TelosB Mote TPR2420 is an open-source platform designed to enable cutting-edge experimentation for the research community. The TPR2420 bundles all the essentials for lab studies into a single platform including: USB programming capability, an IEEE 802.15.4 radio with integrated antenna, a low-power MCU with extended memory and an optional sensor suite. TPR2420 offers many features, including:

- IEEE 802.15.4 compliant RF transceiver
- Temp./ Humidity Sensor
- Light Sensor
- 2.4 to 2.4835 GHz, a globally compatible ISM band
- 250 kbps data rate
- Integrated onboard antenna
- 8 MHz TI MSP430 microcontroller with 10kB RAM
- Low current consumption

- 1MB external flash for data logging
- Programming and data collection via USB
- Sensor suite including integrated light, temperature and humidity sensor
- Runs TinyOS 1.1.11 or higher
- The TelosB platform was developed and published to the research community by UC Berkeley. This platform delivers low power consumption allowing for long battery life as well as fast wakeup from sleep state. The TPR2420 is compatible with the open-source TinyOS distribution.

TPR2420 is powered by two AA batteries. If the TPR2420 is plugged into the USB port for programming or communication, power is provided from the host computer. If the TPR2420 is always attached to the USB port no battery pack is needed.

TPR2420 provides users with the capability to interface with additional devices. The two expansion connectors and onboard jumpers may be configured to control analog sensors, digital peripherals and LCD displays.

TinyOS is a small, open-source, energy-efficient software operating system developed by UC Berkeley which supports large scale, self-configuring sensor networks. The source code software development tools are publicly available at: <http://www.tinyos.net>

4. Software Description

General Requirement: A Laptop loaded with a linux operating system (Ubuntu most preferable)

Refer the Appendix A to know how to install tiny OS.

Tiny OS 2.1.0 was used.

The TelosB mote is configured by developing the codes using Tiny OS. These codes are written, compiled and downloaded into TelosB mote directly by plugging it into USB interface. The code on the memory of TelosB mote can be erased and a different program for a different operation can be downloaded again.

The programs were written and used for the following operations :

1. To provide Start Signal and use mote as a Remote.
2. To collect the data samples (two components of the magnetic field) collected by the magnetometer from the data collection scene and store in the on-chip memory at 100Hz, Sampling Rate.
3. To read the TelosB mote which is connected to the laptop after data acquisition.
4. To transfer the data samples stored in the on-chip memory, to the laptop and save the samples in a text file.

These source codes are present in the directory: Source codes. These codes are developed by SSP lab, ECE dept., IISc.

5. Vehicle Detection

The first important requirement for a traffic surveillance system is the capability to detect the presence of a vehicle. Based on such detection, statistics such as vehicle count, traffic flow speed and occupancy can be estimated. The quality of vehicle detection determines the performance of all dependent applications. The analysis of signals from a magnetic sensor are presented in later chapters.

5.1 Magnetic Signal Analysis

The theory of the Anisotropic Magnetoresistive (AMR) sensor is discussed in [2.1]. It is noted that the earth's magnetic field is uniform over a wide area on the scale of kilometres and that an AMR sensor is capable of detecting distortions in the field. A permeable object in a homogenous magnetic field gives rise to an induced magnetic field. Therefore, when a vehicle enters the vicinity of the sensor, the perturbation of the field can be detected by the sensor. This is the basic principle involved in using an AMR sensor for vehicle detection, see Figure 5.1 .

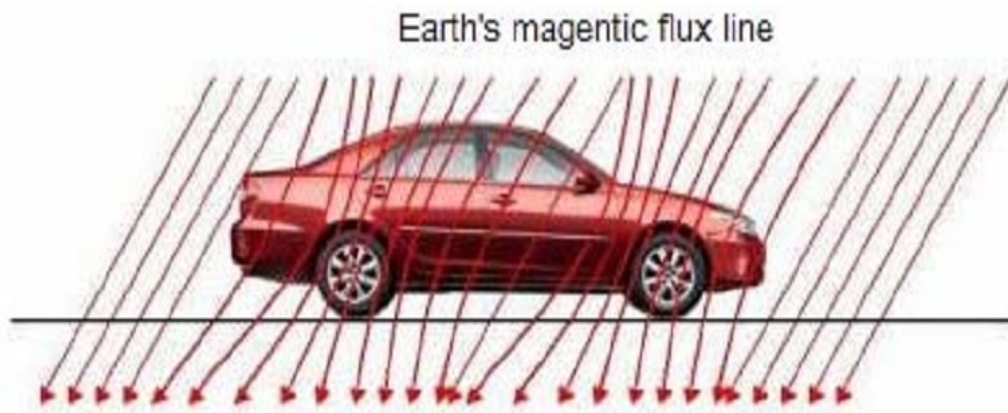


Figure 5.1: Distortion in the earth's magnetic field caused by a vehicle.

Depending on the metallic configuration and composition inside a vehicle, different magnetic signatures are induced by different vehicles.

5.2 Data Collection

The TelosB mote is to be placed in the road condition (middle of lane or side lanes) to sense the magnetic field components. The TelosB mote is having SBT80 board connected to it, which has dual axis magnetometer (HMC1052) But this requires us to ensure that the vehicle is moving at a constant velocity over the mote and the proper care is also taken in protecting the mote from being run-over by the vehicles and getting damaged. The setup in actual road scenario is shown in the Figure 5.2.

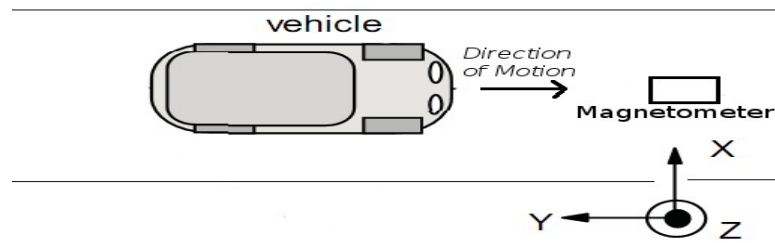


Figure 5.2: Magnetometer setup in actual road scenario

5.3 Modified Setup Similar to Actual Road Scenario

The magnetometer records the magnetic field components whenever the flux lines associated with it is cut by a Vehicle. Essentially there must be a relative motion between the vehicle and the magnetometer.

Owing to the difficulties in the actual road scenario, the data collection is done in parking lots, where the vehicles are stationary. For satisfying the criterion of relative motion the sensor is set in motion. In some ways the conditions of constant velocity and protection of mote are ensured.

The experiments are carried out by fixing the magnetometer setup (TelosB mote) to a Remote Controlled Car and in some cases to a Skate Board.

5.3.1 Experimentation using Remote Controlled Car

A remote controlled car is mounted with the TelosB mote to which SBT80 sensor board is plugged. The mounting is done using a fiber case which encapsulates the mote. Experiment is done and it is ensured that the fiber case does not influence the magnetometer readings (Refer 5.3.2.1). Figures 5.3.1 & 5.3.2 show the photographs of the setup. The orientation of the magnetometer is such that it records the Y and Z-components of the magnetic field.

The specifications of the remote controlled car used is as follows:

working frequency: 49 Mhz

Battery type: 9v pluggin rechargeable, Ni-Cd battery

max. Speed: 32 kmph

The remote control car is having least number of metallic components unlike other conventional remote controlled cars.

The dimension of the remote controlled car is as follows:

height: 2.8"

length: 15.5"

width: 7.5"

height of the fiber box having mote within: 2". Therefore, total height: 4.8"

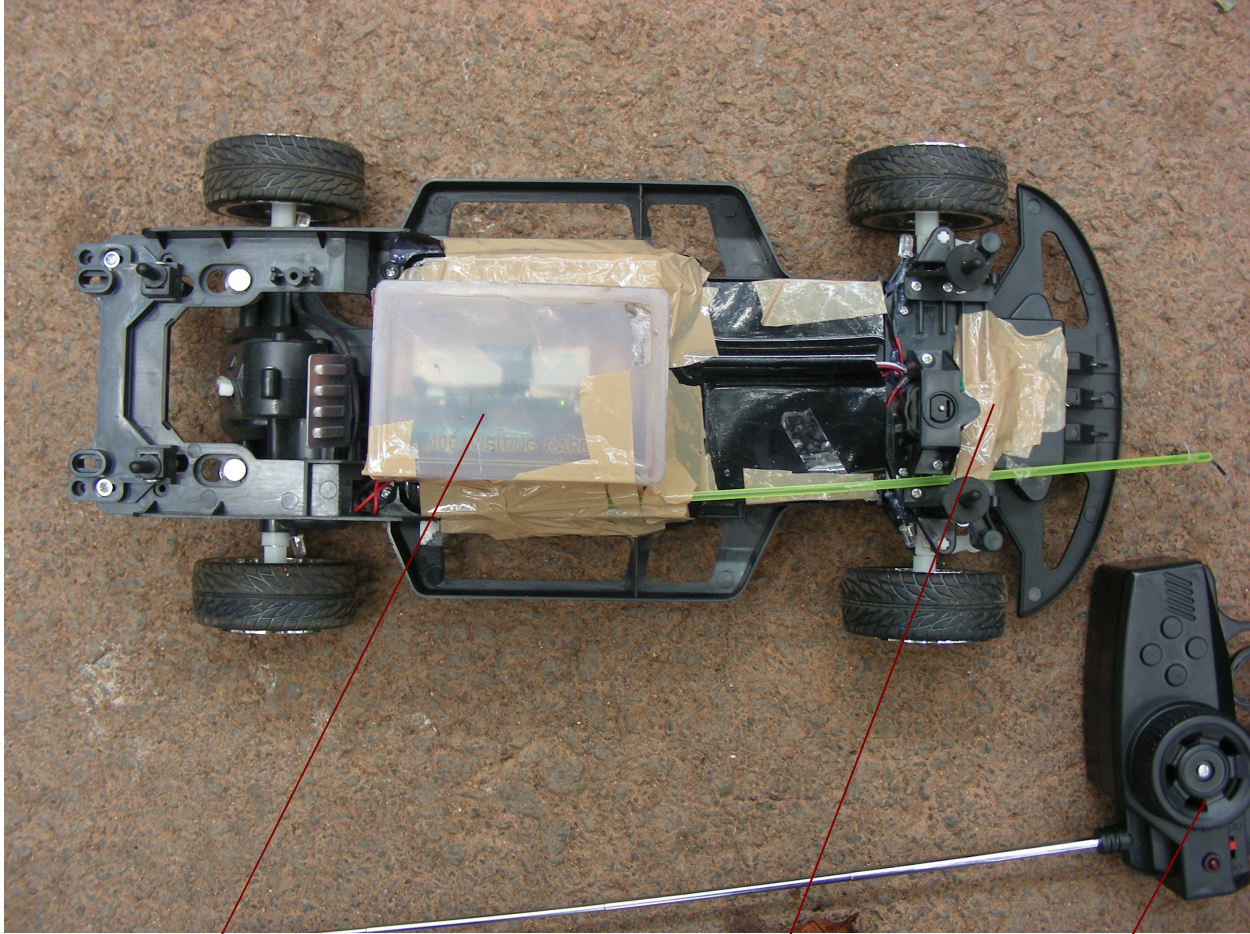


Figure 5.3.1: Top view of Remote Controlled Car

TelosB mote inside fiber case

Fixed Direction controller

Remote

The remote controlled car as shown in the above figure is mounted with the fiber case and mote using double sided gumtapes and brown tapes. The direction controller is fixed so that the setup moves in only straight lines. The motor of the remote controlled car is modified, so that the car moves with constant speed once it overcomes initial acceleration.

Note: The TelosB mote is oriented inside the fiber box in such a direction so that it records Y and Z-components of the magnetic field.



Figure 5.3.2: Side View of the Remote Controlled Car

Antenna of the Remote Car

The remote controlled car having the mote is passed below the vehicles in three different paths viz., left path, middle path & right path as shown in Figure 5.3.3.

A data collection sample video is also taken to provide better understanding for the reader.



Figure 5.3.3: Three paths in which setup is passed

Using the remote the remote controlled car is passed in these 3 paths. The remote car is kept away from the vehicle's influence (say 11 inches in front), it is turned on, mote is turned on, start signal is given in the other mote (mote A) configured for sending out start signal wirelessly.

Now the mote on the remote controlled car (mote B) is left undisturbed for a few seconds so that it records the earth's constant magnetic field. Then the setup is moved in the path using the remote control so that magnetometer records distortions in field. After it covers entire vehicle, the remote controlled car is stopped at considerable distance from the vehicle. The mote B is again left undisturbed as in the beginning.

The data collected is transferred into the laptop and the mote is again configured for a new data collection.

5.3.2 Experimentation using a Skate Board

Parking lots are often having pavement stones and uneven flooring which cause the remote controlled car to drift, changing the velocity of the setup. This can produce errors. A Skate Board can be used in such cases. It has the wheels which have got a provision to move only in linear directions, unlike a remote controlled car. The direction is fixed. If care is taken to move the Skate Board at constant speed, then the Skate Board can serve the same purpose of remote controlled car.

One end of the Skate Board is tied firmly with a long plastic string (rope). A fiber case encapsulating mote B, is fixed to the skate board using double sided gum tapes. The orientation of the magnetometer is such that it records the Y and Z-components of the magnetic field.

The dimensions of the skate board is as follows:

height: 4"

length: 23.5"

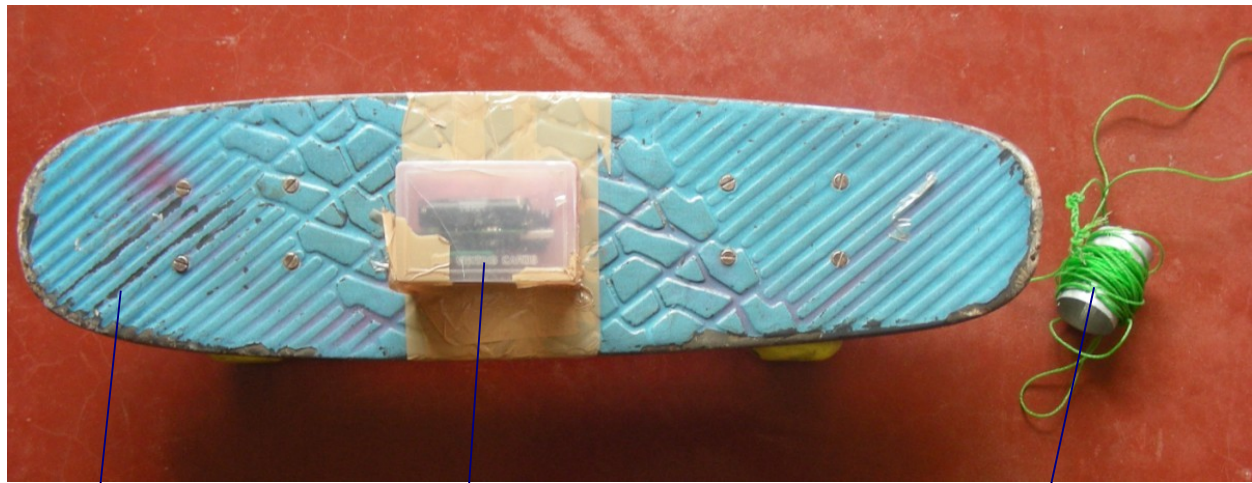
width: 6"

height of the fiber box: 2"

Therefore total height when fiber box is mounted: 6"

In certain cases it was realised that the ground clearance of the vehicle $< 6"$

In such cases the fiber casing was fixed to the bottom face of the Skate Board. An experiment was also conducted to see if the position of the mote when fixed to top/bottom face influence the readings of the magnetometer (Refer 5.3.2.2). Based on the results of this experiment, Skate Board could be used in both the ways. Figures 5.3.4 to Figure 5.3.7 show the top views and side views of the setup used.



Top face of skate board

TelosB mote inside fiber case

Roll of plastic string tied to one end

Figure 5.3.4: Top view of setup mounted on the Skate Board

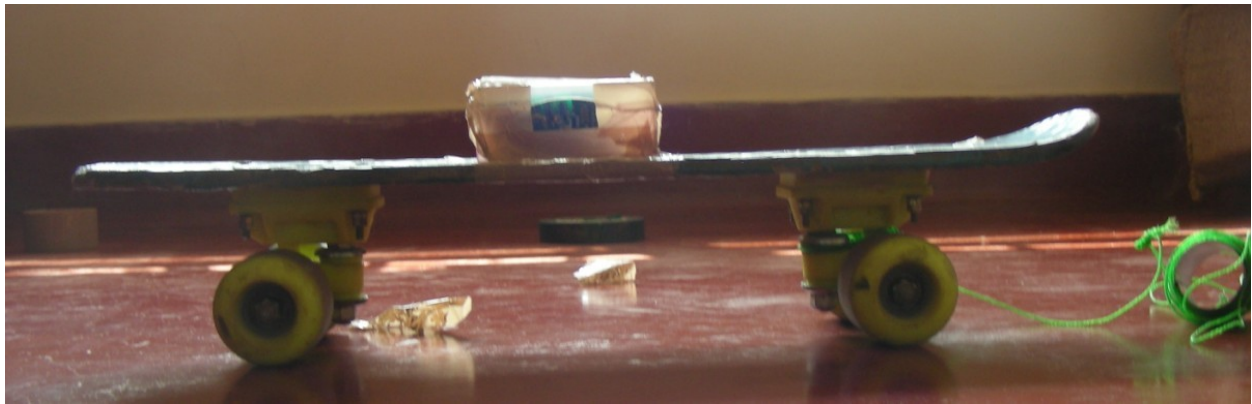


Figure 5.3.5: Side view of setup mounted on the Skate Board

This setup is used in the same way like the remote controlled car which was used before. The same procedure is followed to record the magnetic field components. Care is taken to pull the setup with constant velocity. A data collection sample video is also taken to provide better understanding for the reader.

Note: The TelosB mote is oriented inside the fiber box in such a direction so that it records Y and Z-components of the magnetic field.

5.3.2.1 Experiment no. 1

Aim: To study the effect when a Telosb mote is enclosed within a fiber box.

Explanation: In this experiment, the Telosb mote is mounted on a skate board without any enclosure. The readings are taken using this arrangement by powering the mote. But we have to protect the Telosb mote from external factors like rain, dust and also possible damage due to insufficient clearance midway through the vehicle, during our experiments. Hence we first enclose the Telosb mote within a fiber case and then carry out the experimentations, as usual. In order to do so, we must study the effect on the magnetic field when the Telosb mote is enclosed within the fiber case.

Experimental setup:

Case1- The Telosb mote is mounted on the skate board without any enclosure. The mote was powered and the entire set-up was rotated in an anticlockwise direction, twice. The samples, thus obtained were plotted in the Matlab. The plot is given below:

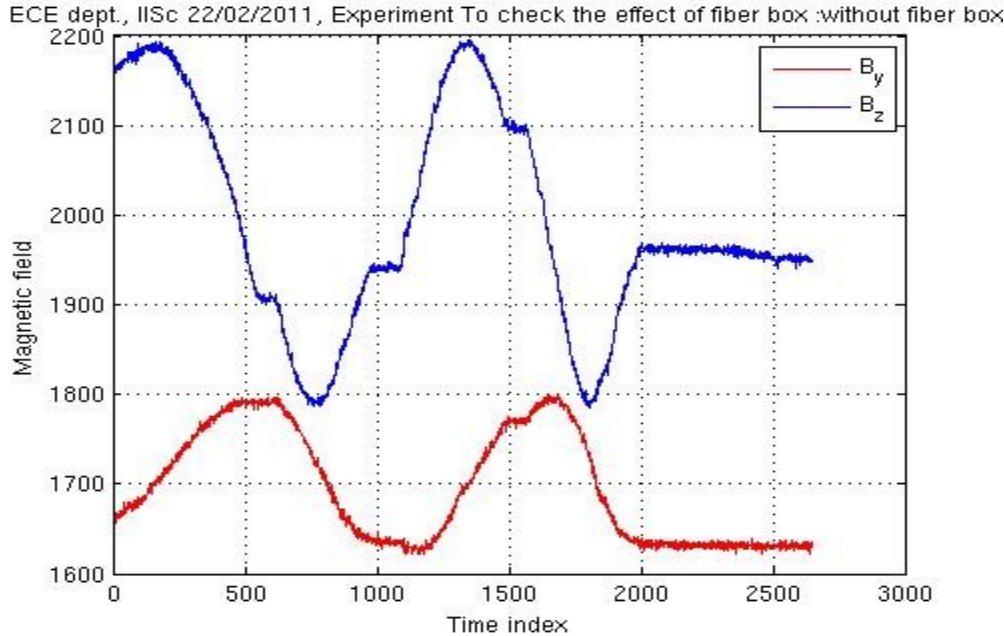


Figure 5.3.6: Waveforms obtained without fiber box

The peak to peak change in the Magnetic field observed from the above figure is:

2195-1795=400 units.

Case 2- The Telosb mote is first enclosed inside a fiber case with cover and then mounted on top of the skate board. The mote was powered and the entire set-up was rotated in an anticlockwise direction, twice (similar to case2). The samples, thus obtained were plotted in the Matlab. The plot is given below:

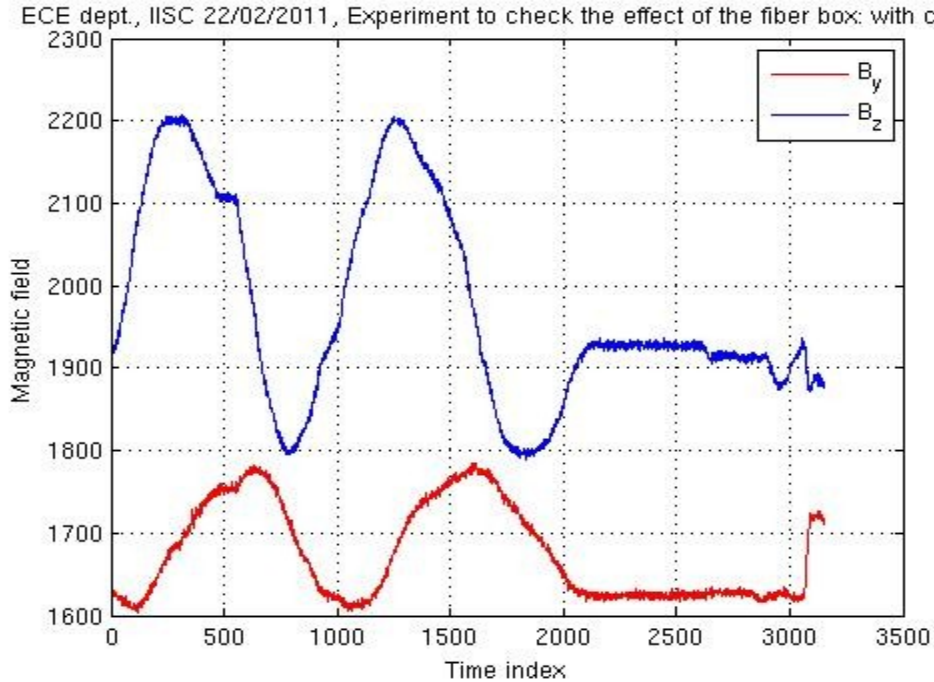


Figure 5.3.7: Waveforms obtained with fiber box

The peak to peak change in the Magnetic field observed from the above figure is:

2200-1798=402 units.

Inference: From the above plots and observations, it is clear that there is “no appreciable change” in the peak-peak magnetic field. Thus we can state that both the experimental cases yield the same result. Hence, owing to the fact that we get added protection and also negligible change in the magnetic field when we enclose the mote within the fiber case, we continue to do so in our further experimentations.

5.3.2.2 Experiment no. 2

Aim: To study the effects when a Telosb Mote is mounted to top face and fixed to bottom face of the skate board.

Explanation: When the Telosb mote is mounted on the skate board, it is practically very difficult to get enough road clearance for few of the vehicles like Hyundai Verna, Ford figo under test because of the additional height the mote brings in. Hence the idea is to fix the mote to the bottom face of the skate board so that we can negate the additional height of the mote. When such a practical consideration is put to use, the objective of getting enough road clearance is achieved. The next concern is whether this type of arrangement would have any effect on the readings. Hence an experiment is conducted to throw light on this practical aspect of examining whether both the arrangements yield similar results or not?

Experimental setup:

Case1- The Telosb mote is mounted on top face of the skate board. The mote is powered and the entire set-up is rotated in an anticlockwise direction, once. The samples, thus obtained are plotted in the Matlab. The plot is given below:

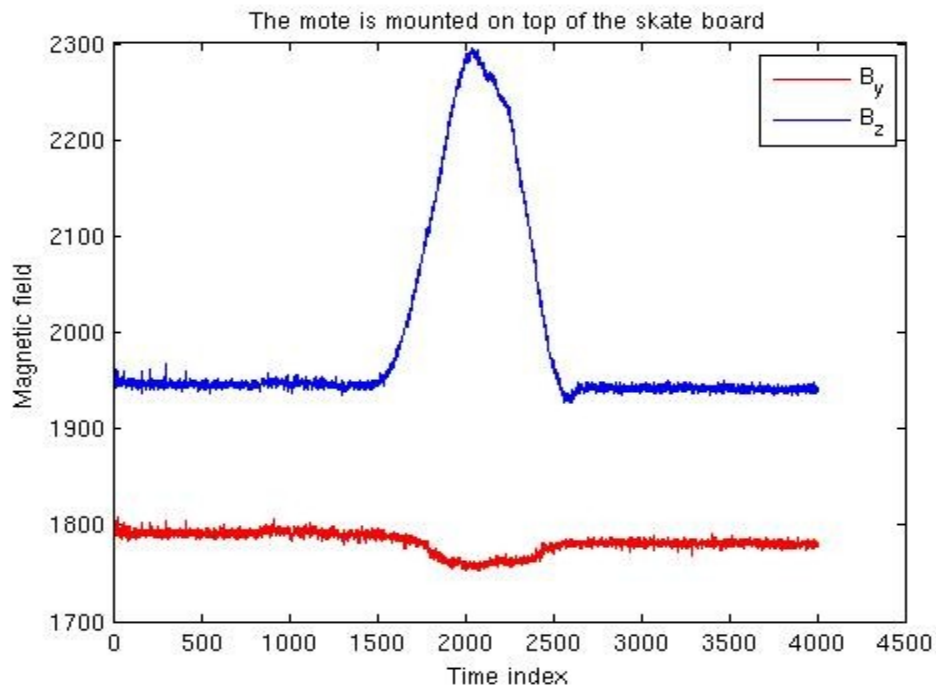


Figure 5.3.8: Waveforms obtained when mote is mounted on Skate Board

The peak to peak change in the Magnetic field observed from the above figure is:

$$2295-1927=368 \text{ units.}$$

Case 2- The Telosb mote is fixed to bottom face of the skate board. The mote is powered and the entire set-up is rotated in an anticlockwise direction, once (exactly similar to Case1). The samples, thus obtained are plotted in the Matlab. The plot is given below:

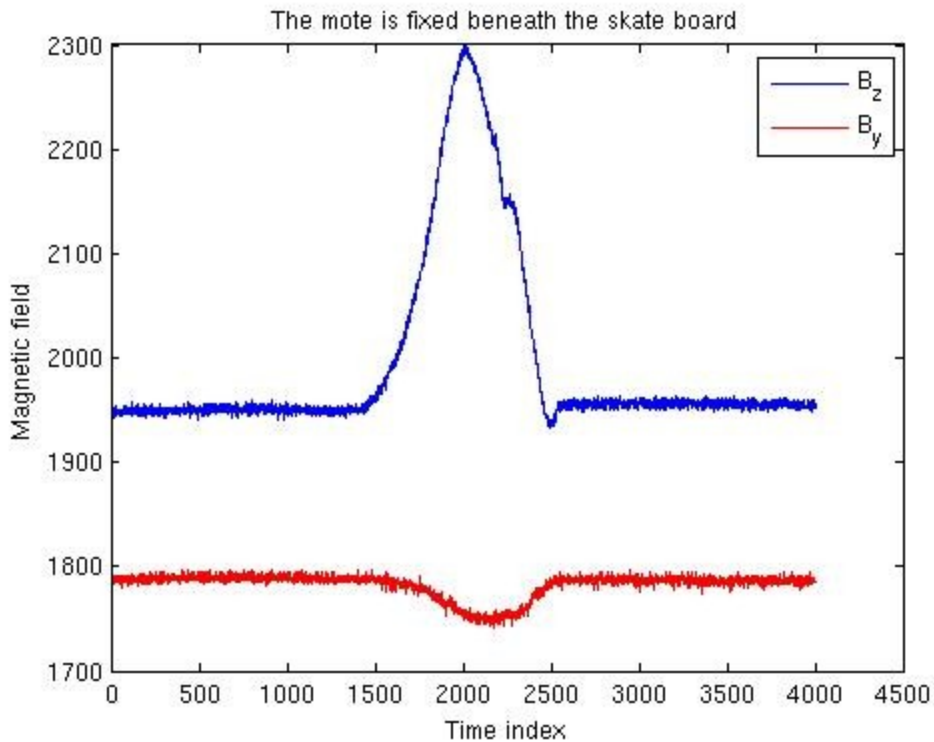


Figure 5.3.9: Waveforms obtained when mote is fixed to bottom face of Skate Board

The peak to peak change in the Magnetic field observed from the above figure is:

$$2299-1932=367 \text{ units.}$$

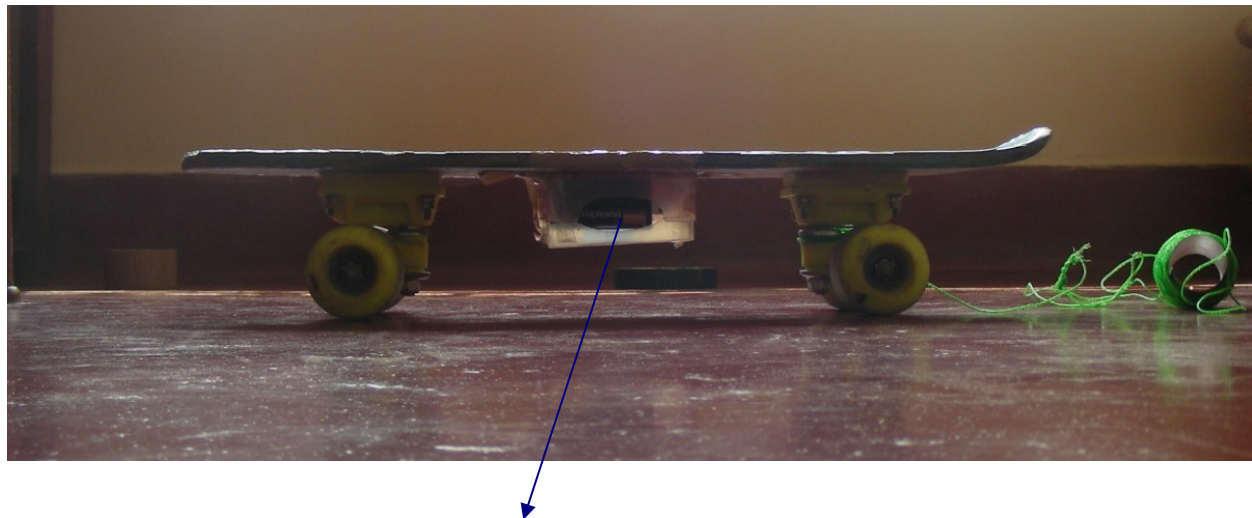
Inference: From the above plots and observations, it is clear that there is “no appreciable change” in the peak-peak magnetic field. Thus we can state that both the experimental cases yield the same result. Hence, owing to the fact that we get clearance as well as protection and also negligible change in the magnetic field when we fix the mote to bottom face of the skate board, we continue to do so in our further experimentations.



Bottom face of skate board

String tied firmly to the wheel frame

Figure 5.3.10: Top view of setup fixed to the bottom face of the Skate Board



TelosB mote inside fiber box

Figure 5.3.11: Side view of setup fixed to the bottom face of the Skate Board

The above setup can pass easily under the cars having less ground clearance. The experiments were performed with the above mentioned setups and following vehicles were covered for extracting the signature signals. The general procedure to configure the TelosB motes is given in the Appendix B. The table 5.3 shows all cars covered by using the two setups.

Table 5.3: All cars covered using the two setups

Sl. no	Name of the Car	Number of cars covered
1.	Maruti 800	6
2.	Maruti 800(old)	2
3.	Maruti Suzuki Alto	2
4.	Maruti Omni	4
5.	Maruti Omni(old)	2
6.	Maruti Suzuki Swift	2
7.	Maruti Suzuki Swift Dzire	1
8.	Maruti Suzuki Zen	2
9.	Maruti Suzuki Zen Estilo	3
10.	Maruti Suzuki WagonR	3
11.	Maruti Suzuki WagonR(duo)	1
12.	Maruti Suzuki Sx4	2

13.	Maruti Suzuki Ritz	1
14.	Maruti Esteem	2
15.	Hyundai Santro	2
16.	Hyundai Santro Xing	3
17.	Hyundai Accent	1
18.	Hyundai Elantra	2
19.	Hyundai i10	4
20.	Hyundai i20	1
21.	Hyundai Verna	1
22.	Hyundai Getz	2
23.	Hyundai Sonata	1
24.	Tata Indica	6
25.	Tata Indigo	2

26.	Tata Nano	2
27.	Tata Sumo	1
28.	Volkswagen Vento	1
29.	Bajaj Auto	1
30.	Chevrolet Beat	2
31.	Chevrolet Spark	1
32.	Honda Civic	1
33.	Honda City(new)	3
34.	Honda City(old)	1
35.	Ford Figo	2
36.	Ford Fiesta	1
37.	Daewoo Matiz	3
38.	Daewoo Cielo	1
39.	Toyota Corolla SE	1

40.	Toyota Innova	2
41.	Fiat Linea	1
42.	Fiat Petra	1
43.	Fiat Palio	1
44.	Mahindra Logan	1
45.	Mahindra Jeep	1
46.	Skoda Octavia	2
47.	Skoda Laura	1
48.	Reva Electric Car	1
49.	Opel Corsa	1
50.	Opel Corsa Sail	1

5.4 Organising the data collected into Files/Directories

- Each car among the various cars covered as shown in Table 5.3 is given a directory name based on the name/model of the car.
- This directory contains all the associated files.
- All the associated files are named in the following general order:

<car name>_<order of the car>_<path>.<extension>

For Example: Let us say that we have recorded the magnetic field data samples in three paths namely left, right and middle for a Maruti Suzuki 800. The data samples are found in a text file. Thus, the extension of the files will be .txt.

The car name is most likely to be 800. The order of the car relates to the total number of cars of the same type covered till the particular car. That is, if 2 Maruti Suzuki 800 cars were already covered earlier, then the present car would be given the order 3. In case of car which was covered only once, then <order of the car> is omitted. The path name could be left, middle or right for which the data samples were collected. In some cases the car models are also indicated using new/old. For eg. Honda City, which has old/new models. 800 and Omni also have new/old models. If multiple readings are taken for a same path then, path_trial no. is indicated.

Using the text file for each path, an associated MATLAB figure having .fig extension and a figure of .jpeg extension are extracted. The photograph(s) of the car and difficulties(if any) is/are presented.

A <car name>.doc file gives all the information associated with a particular car in each directory of cars.

These cars are found in directories: Vehicle detection_r_cars or Vehicle detection_s_cars. A file map which gives a picture of the directories is present. Associated with each main directory there is a README.txt which gives the summary of contents/information of each directory.

Two video files each in above mentioned main directories are present in .mov extension.

Appendix A

Installing TinyOS

1. TinyOS on Fedora

The procedure for installing TinyOS on Fedora has been described here.

1. As root, copy the contents of the given **TinyOS_Setup** folder into the **/opt** folder.

2. Go to the **/opt** folder and execute the following command :

sh tinyos-install.sh.

3. Once the execution is complete, the **tinyos-2.x** folder is created in the **opt** folder.

4. Next, go to the folder java :

cd /opt/tinyos-2.x/support/sdk/java

5. In the **java** folder, execute the make command : **make**

6. This completes the installation of TinyOS.

2. TinyOS on Ubuntu

2.1 Default Version

The procedure for installing the default TinyOS version on Ubuntu 8.10 has been described here. A similar procedure can be used for installing TinyOS on other versions of Ubuntu.

1. Add the repository for TinyOS to the repository cache. Open the file **/etc/apt/sources.list** as sudo and

append the following lines to it :

#repository for TinyOS

deb http://tinyos.stanford.edu/tinyos/dists/ubuntu feisty main

2. Update the repository cache. Run the following command :

sudo apt-get update

3. **Note:** If you have an older version of TinyOS installed, remove it:

sudo apt-get remove tinyos

and then proceed.

4. Run the following command to install the latest release of TinyOS and all its supported tools:

sudo apt-get install tinyos

This will display a message asking you to choose between available versions. Select the appropriate version.

For example :

sudo apt-get install tinyos-2.1.0

5. Since we are using telosb motes, we need to uninstall the braille drivers:

sudo apt-get remove brltty

6. We need to edit the ~/.bashrc file to set environment variables and define aliases.

- Open the **tinyos-bashrc** file present in the TinyOS Setup folder.
- Change all occurrences of **tinyos-2.x** to **tinyos-2.1.0**.
- Change the CLASSPATH env-variable as below:

CLASSPATH=\$CLASSPATH:\$TOSROOT/support/sdk/java/tinyos.jar:.

(The . included!)

- Make sure that the env variable TOSROOT is:

TOSROOT="/opt/tinyos-2.1.0"

- Now, copy the contents of this file and append it to the ~/.bashrc file.

7. To the bashrc file, add the following code snippet:

```
if [ -f /opt/tinyos-2.1.0/tinyos.sh ] ; then
```

```
./opt/tinyos-2.1.0/tinyos.sh
```

```
fi
```

8. Change the ownership of the /opt/tinyos-2.1.0 directory to your username. Run the following command

anywhere outside the /opt/tinyos-2.1.0 directory:

chown -R username /opt/tinyos-2.1.0

9. Change permissions on any serial, usb or parallel devices you are going to use:

chmod 666 /dev/devicename

10. You can use the command **motelist** to know where your mote is connected.

11. Execute the following command in the tinyos-2.1.0 directory:

tos-check-env

Ignore the warnings for java or graphviz and proceed.

12. Now execute:

cd /opt/tinyos-2.1.0/tos/lib/tosboot

sudo make telosb

It should compile without errors. If it doesn't, go through all the previous steps carefully and re-execute the command.

13. Now execute:

cd /opt/tinyos-2.1.0/support/sdk/java

sudo make all

If there are errors displayed concerning JNI libraries, then generally the problem is the presence of multiple installations of java.

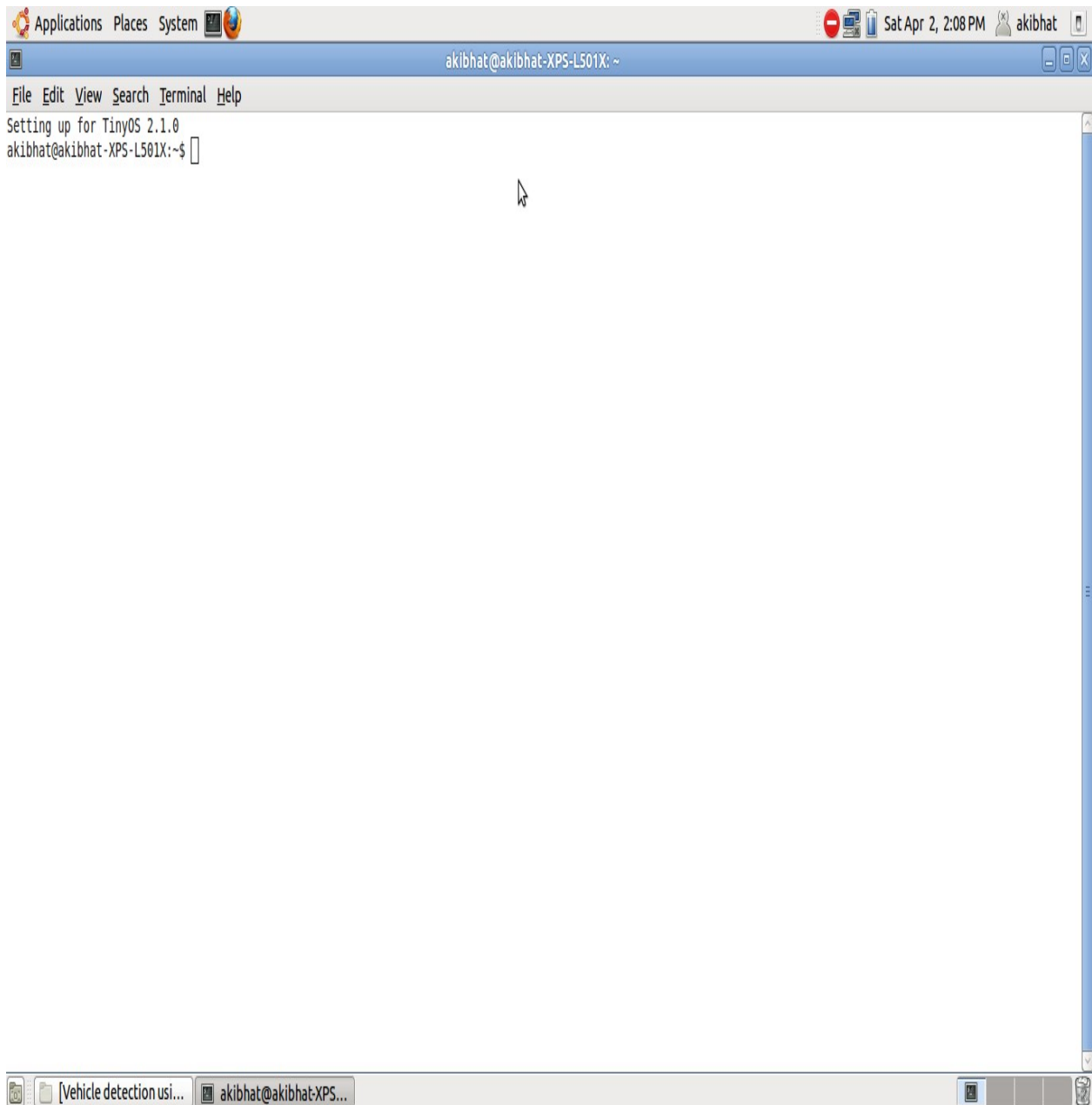
Appendix B

Steps to be followed for configuring TelosB mote

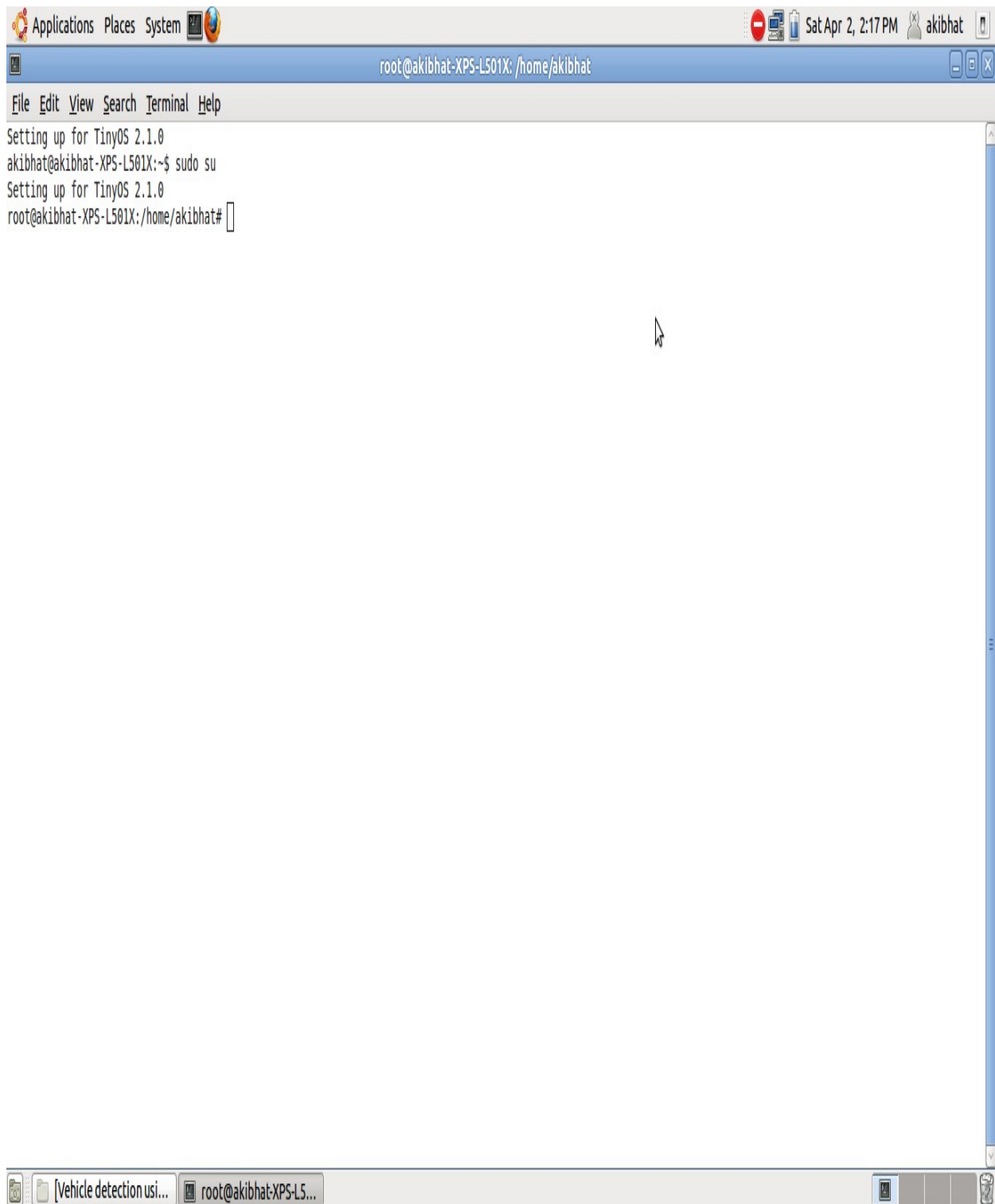
1. To configure TelosB mote (mote A) to send the start signal to moteB

-Plug the mote A to USB interface

-Open **Terminal**



-**Super User** privileges have to be obtained

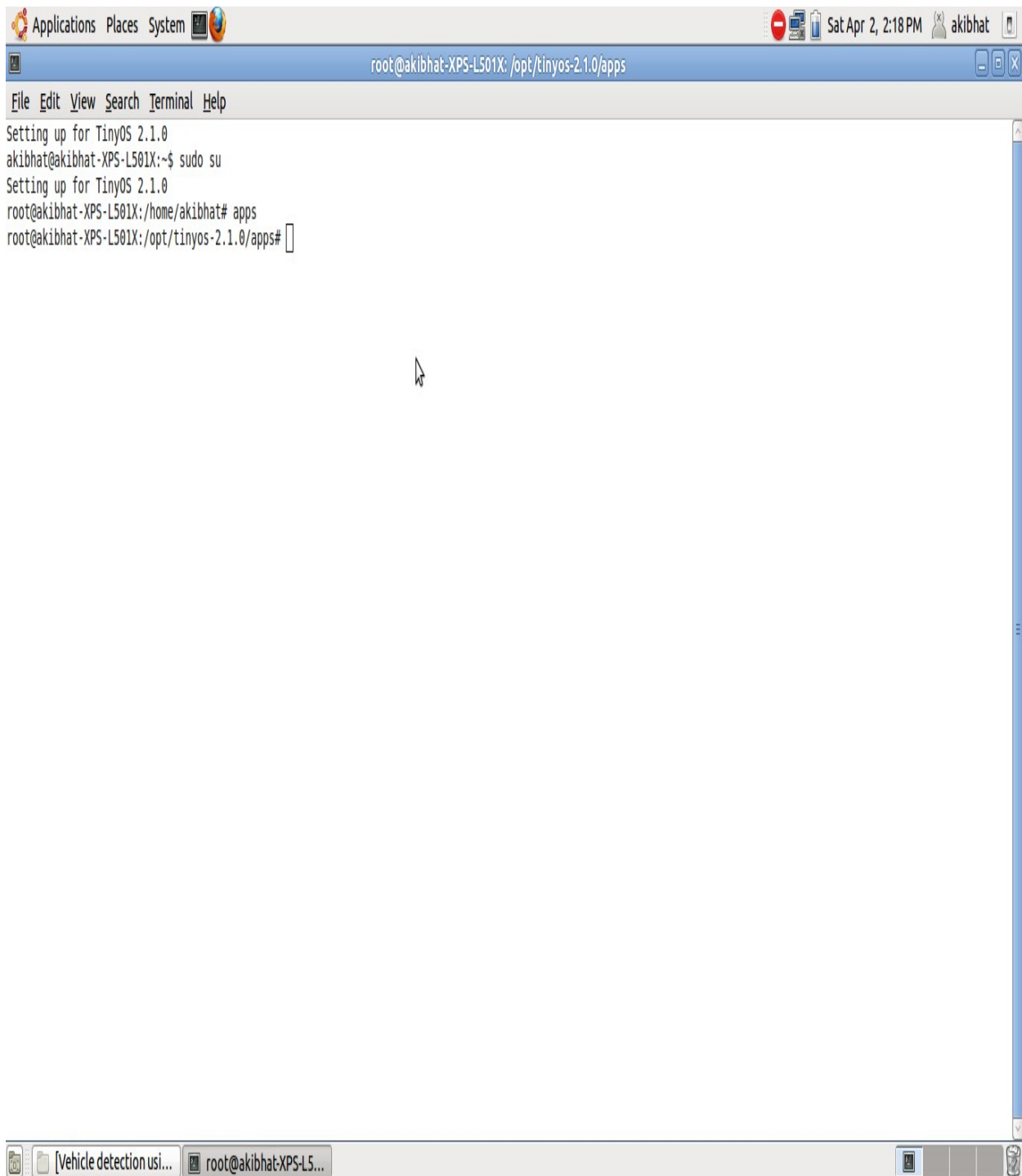


The image shows a Linux desktop environment with a terminal window open. The terminal window has a title bar that reads "root@akibhat-XPS-L501X: /home/akibhat". The terminal content shows the following sequence of commands and output:

```
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat#
```

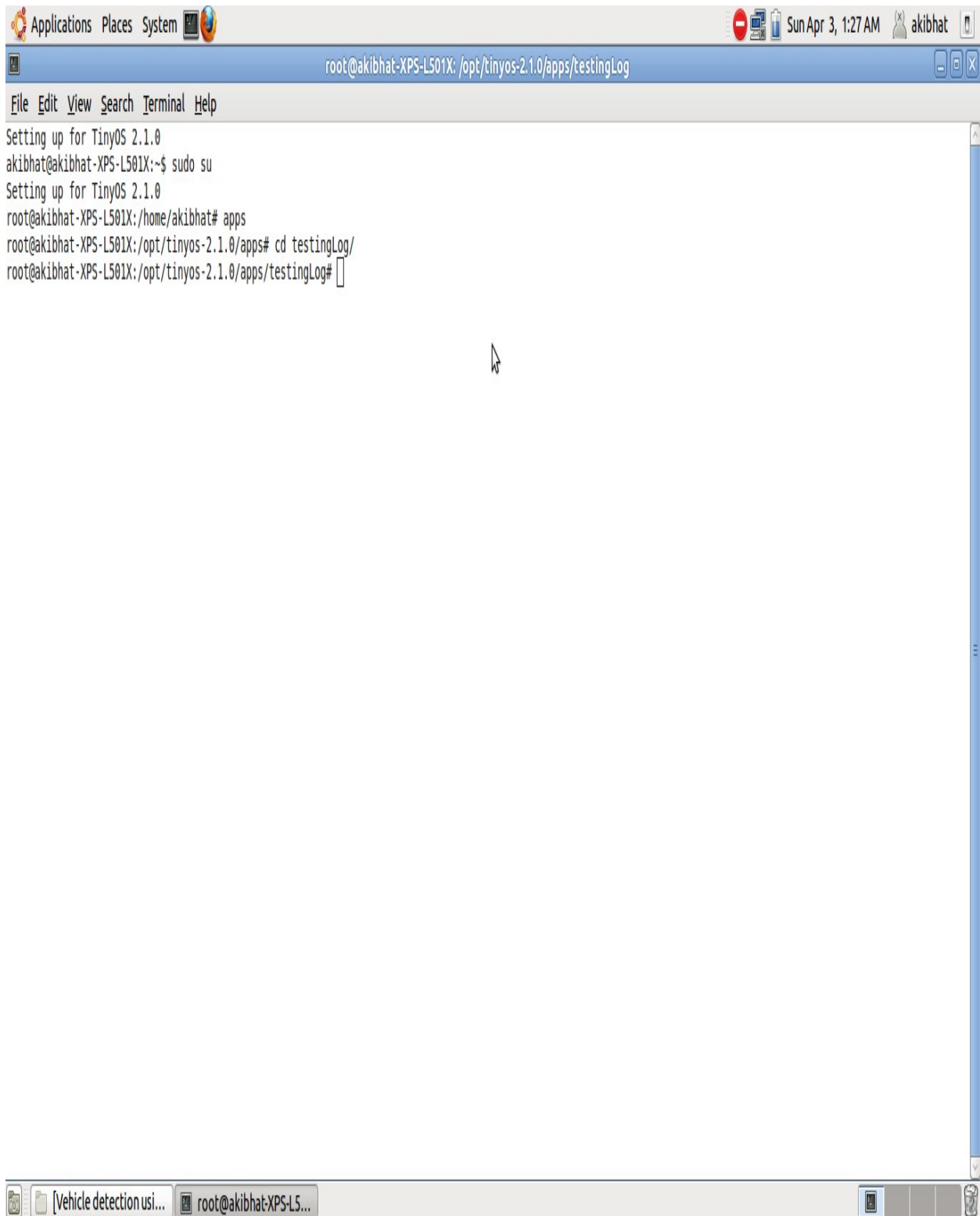
The terminal window is part of a desktop environment with a menu bar at the top containing "Applications", "Places", and "System". The system tray at the bottom right shows the date and time as "Sat Apr 2, 2:17 PM" and the username "akibhat". The taskbar at the bottom shows two open windows: "[Vehicle detection usi..." and "root@akibhat-XPS-L5...".

-Type **apps** (To enter into the directory in Tiny OS)



-Enter the directory **testingLog**.

-Type **cd testingLog/**



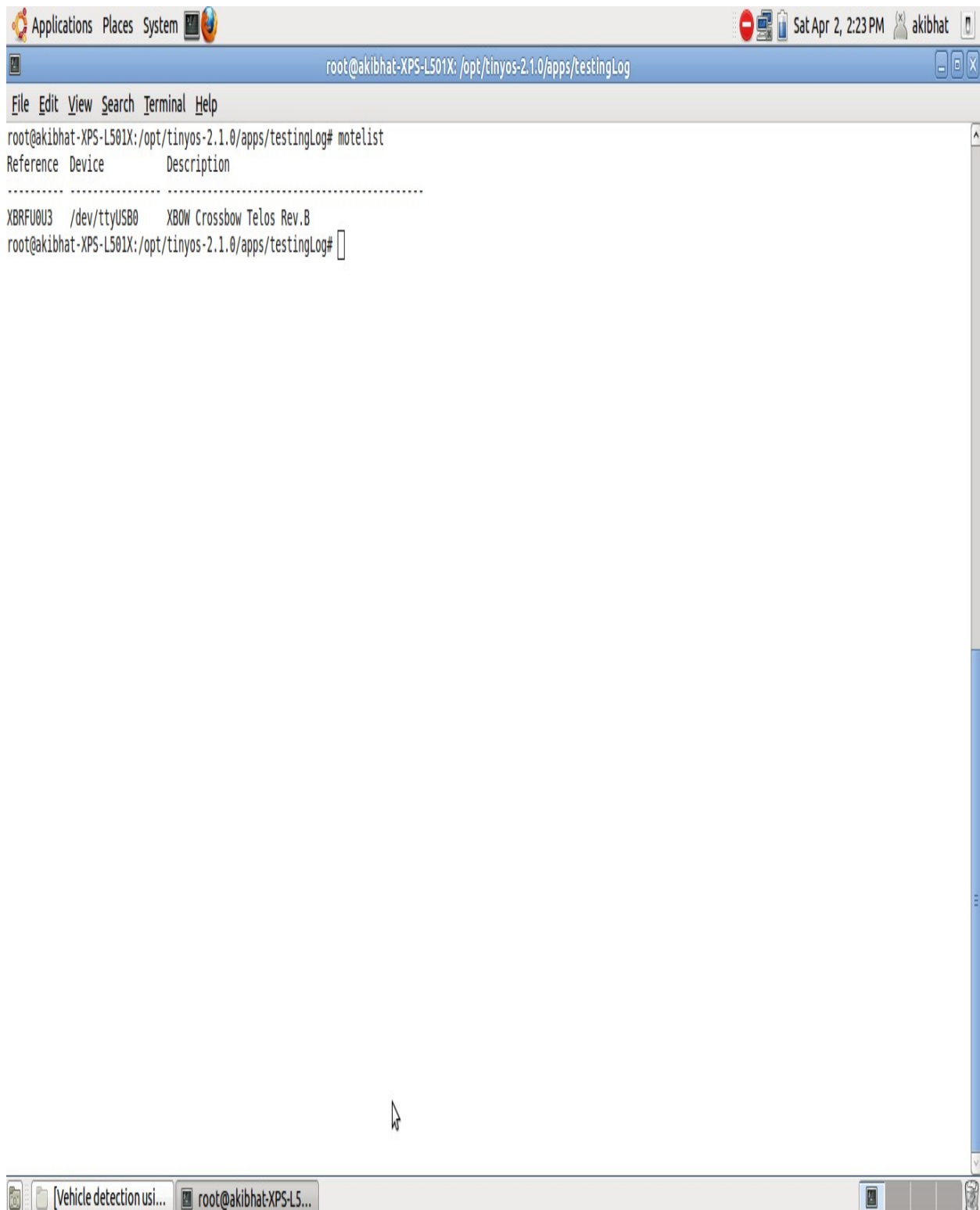
The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar that reads "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog". The terminal content shows the following sequence of commands and output:

```
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog#
```

The terminal window is part of a desktop environment with a top panel showing "Applications Places System" and a date/time display of "Sun Apr 3, 1:27 AM". The bottom panel shows a taskbar with a "Vehicle detection usi..." window and the terminal window.

-Type **motelist** (To check if the mote is connected to USB interface)

The following screen appears if its connected properly.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar reads 'root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog'. The terminal content shows the command 'motelist' being executed, which outputs a table of connected devices. The table has two columns: 'Reference' and 'Device', with a 'Description' header. The output shows a single device: 'XBOW Crossbow Telos Rev.B' connected at '/dev/ttyUSB0'. The terminal prompt is 'root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog# '.

```
root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog# motelist
Reference Device      Description
-----
XBOW Crossbow Telos Rev.B
/dev/ttyUSB0
```

-Enter into the directory **StartS/**

-Type **cd StartS/**




The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar reads "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/StartS". The terminal content shows the following sequence of commands and outputs:

```
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd trLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/trLog# cd ..
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd StartS/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/StartS#
```

The desktop background is light blue. The top panel shows "Applications Places System" menus and system status icons. The bottom panel shows a taskbar with a "Vehicle detection usi..." window and the terminal window.

-Type **make telosb install.0** (To write the code StartS into the mote A)



```
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# clear
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd StartS/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/StartS# make telosb ins
tall.0
mkdir -p build/telosb
  compiling StartSAppC to a telosb binary
ncc -o build/telosb/main.exe -Os -O -mdisable-hwmul -Wall -Wshadow -Wnesc-all -
target=telosb -fnesc-cfile=build/telosb/app.c -board= -DDEFINED_TOS_AM_GROUP=0x2
2 -DIDENT_APPNAME=\"StartSAppC\" -DIDENT_USERNAME=\"root\" -DIDENT_HOSTNAME=\"ak
ibhat-XPS-L501X\" -DIDENT_USERHASH=0xbf0198e2L -DIDENT_TIMESTAMP=0x4d96e6ddL -DIDE
NT_UIDHASH=0xd918ffL StartSAppC.nc -lm
/opt/tinyos-2.1.0/tos/chips/cc2420/lpl/DummyLpLC.nc:39:2: warning: #warning \"***
LOW POWER COMMUNICATIONS DISABLED ***\"
  compiled StartSAppC to build/telosb/main.exe
      11260 bytes in ROM
      322 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 ActiveMessageA
ddressC$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-telos-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 ...
11292 bytes programmed.
Reset device ...
rm -f build/telosb/main.exe.out-0 build/telosb/main.ihex.out-0
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/StartS#
```

Now the mote A is configured and ready to be used for giving Start signal to mote B.

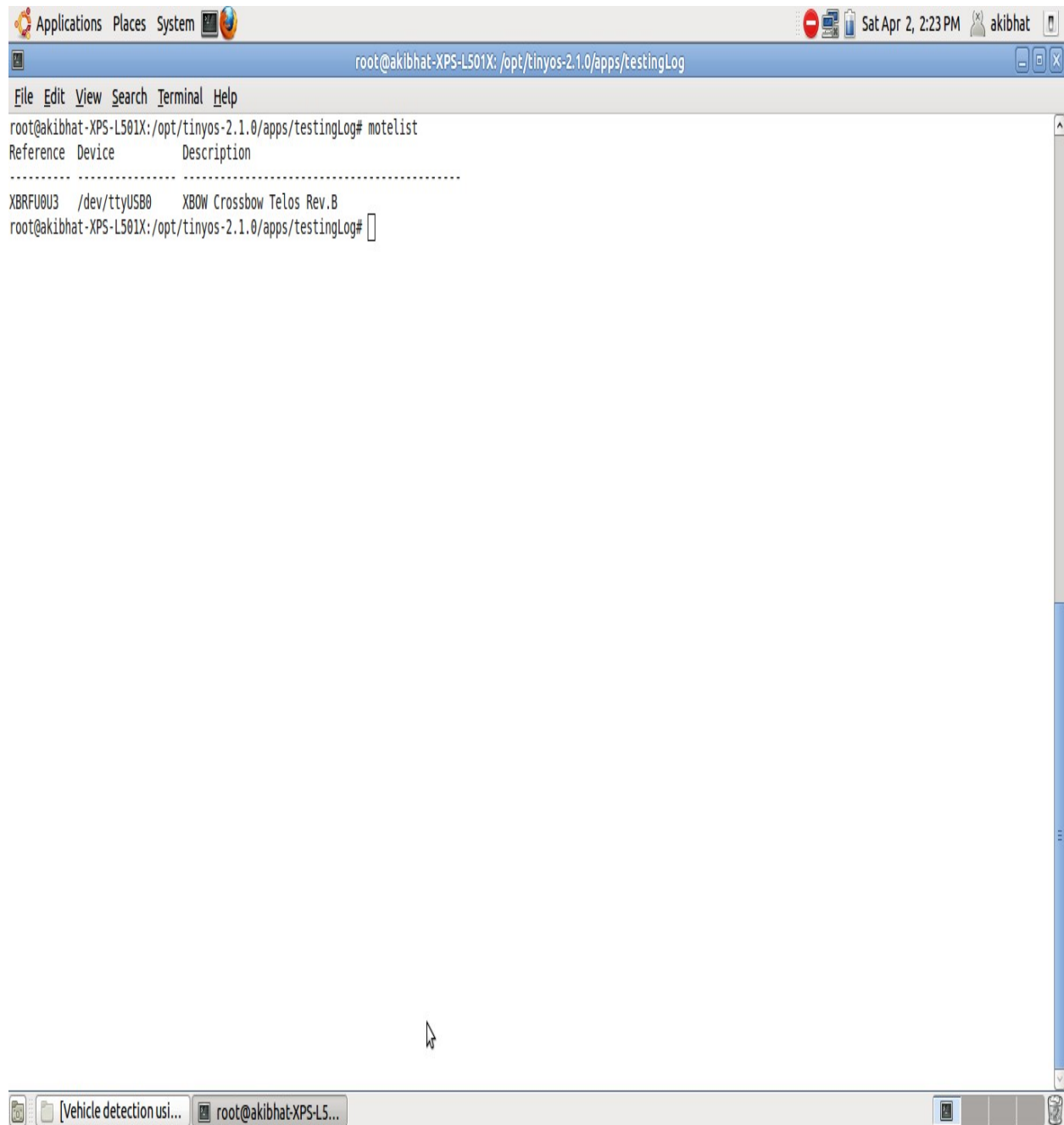
Come out of the directory by typing **cd ..**

2. To configure TelosB mote (mote B) to sense the magnetic field components after it is receives a Start signal from mote B.

-Plug the mote B to USB interface

-Type **motelist** (To check if the mote is connected to the USB interface)

The following screen appears if its connected properly.



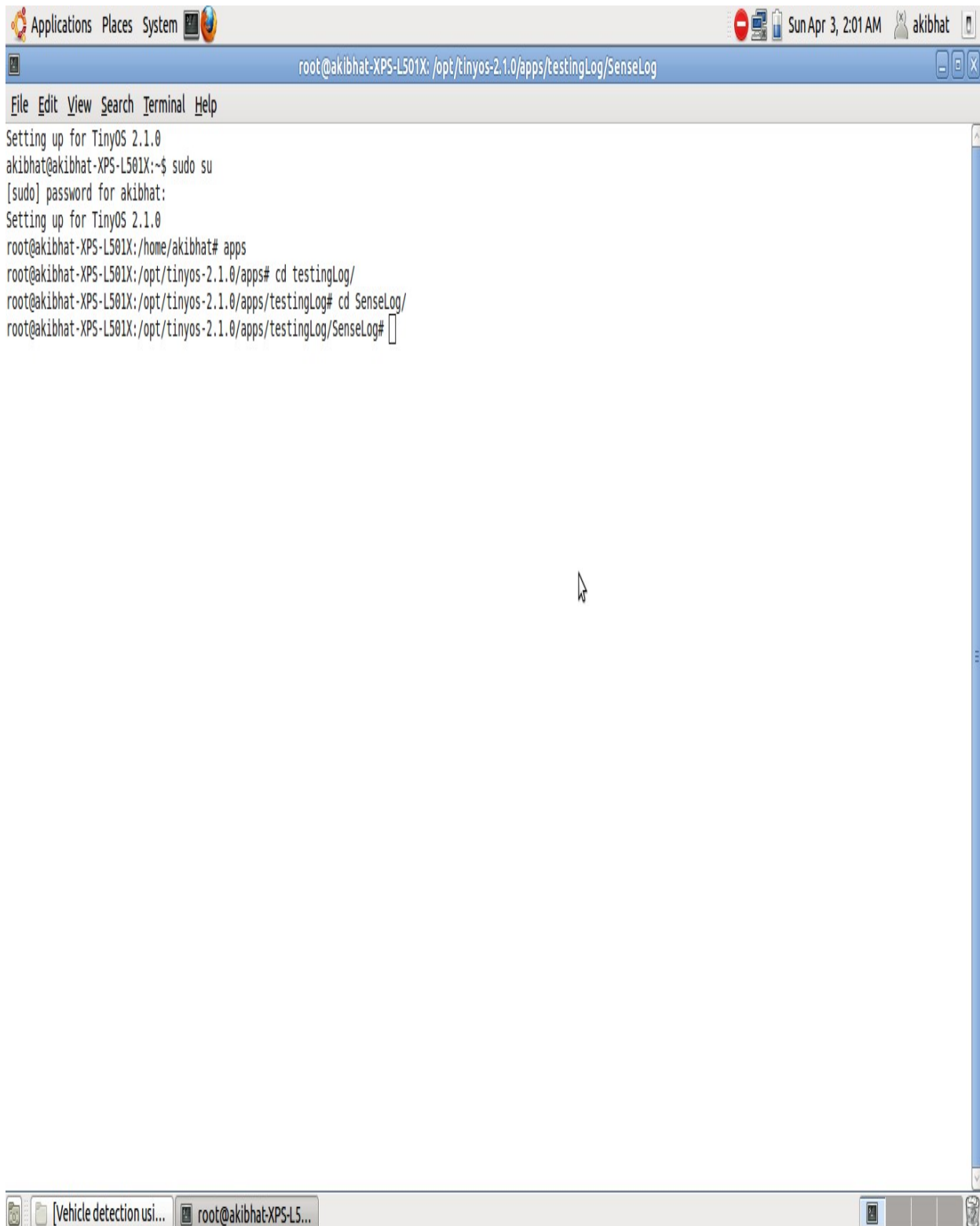
The screenshot shows a terminal window titled "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog". The terminal displays the output of the 'motelist' command, which lists the connected mote. The output is as follows:

```
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# motelist
Reference Device      Description
-----
XBRFU0U3  /dev/ttyUSB0  XBOW Crossbow Telos Rev.B
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog#
```

The terminal window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The status bar at the bottom shows the current directory and the user's name "akibhat".

-Enter into the directory **SenseLog**

-Type **cd SenseLog**



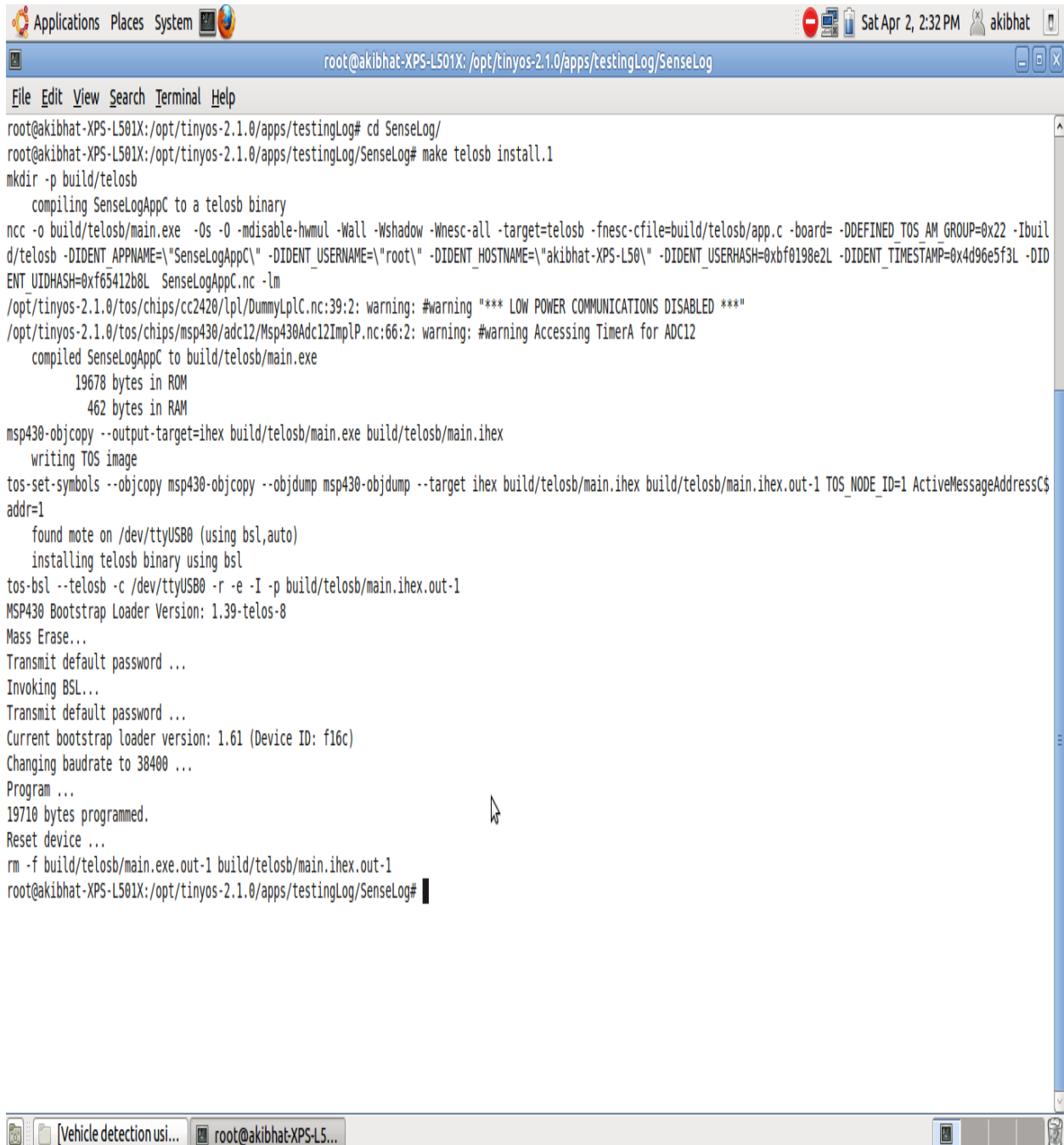
The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar that reads "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/SenseLog". The terminal content shows the following sequence of commands and outputs:

```
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
[sudo] password for akibhat:
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd SenseLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/SenseLog#
```

The desktop environment includes a top panel with icons for Applications, Places, and System, and a system tray showing the date and time as "Sun Apr 3, 2:01 AM" and the user "akibhat". The bottom panel shows a taskbar with a window titled "[Vehicle detection usi..." and another window titled "root@akibhat-XPS-L5...".

-Type **make telosb install.1** (This is to write the code SenseLog into the mote B)

Note: The number index '1' used in **make telosb install.1** user defined. However it must be different from the index used for StartS.



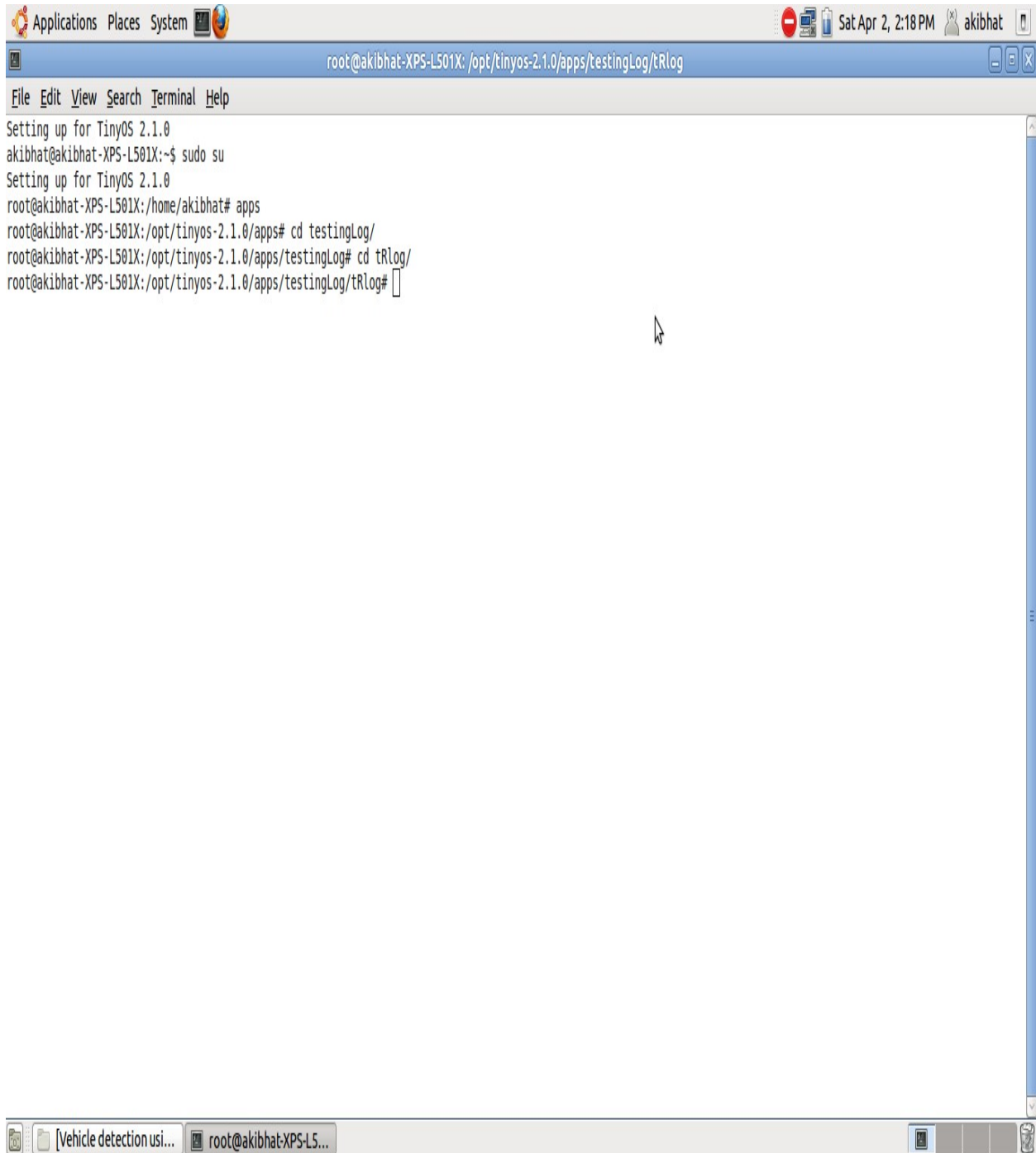
```
root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/SenseLog
File Edit View Search Terminal Help
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd SenseLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/SenseLog# make telosb install.1
mkdir -p build/telosb
  compiling SenseLogAppC to a telosb binary
ncc -o build/telosb/main.exe -Os -O -mdisable-hwmul -Wall -Wshadow -Wnesc-all -target=telosb -fnesc-cfile=build/telosb/app.c -board= -DDEFINED_TOS_AM_GROUP=0x22 -Ibuild/telosb -DIDENT_APPNAME="\SenseLogAppC" -DIDENT_USERNAME="root" -DIDENT_HOSTNAME="akibhat-XPS-L501X" -DIDENT_USERHASH=0xbf0190e2L -DIDENT_TIMESTAMP=0x4d96e5f3L -DIDENT_UIDHASH=0xf65412b8L SenseLogAppC.nc -lm
/opt/tinyos-2.1.0/tos/chips/cc2420/lpl/DummyLplC.nc:39:2: warning: #warning "*** LOW POWER COMMUNICATIONS DISABLED ***"
/opt/tinyos-2.1.0/tos/chips/msp430/adcl2/Msp430Adc12ImplP.nc:66:2: warning: #warning Accessing TimerA for ADC12
  compiled SenseLogAppC to build/telosb/main.exe
      19678 bytes in ROM
      462 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-1 TOS_NODE_ID=1 ActiveMessageAddressC$
addr=1
  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-1
MSP430 Bootstrap Loader Version: 1.39-telos-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 ...
19710 bytes programmed.
Reset device ...
rm -f build/telosb/main.exe.out-1 build/telosb/main.ihex.out-1
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/SenseLog#
```

Now the TelosB mote, mote B is ready to sense the magnetic field components once the Start signal is received.

3. To configure the TelosB mote (mote B) so that data collected by it can be read by a laptop and a text file having the data is generated.

-Enter the directory **tRlog**

-Type **cd tRlog**



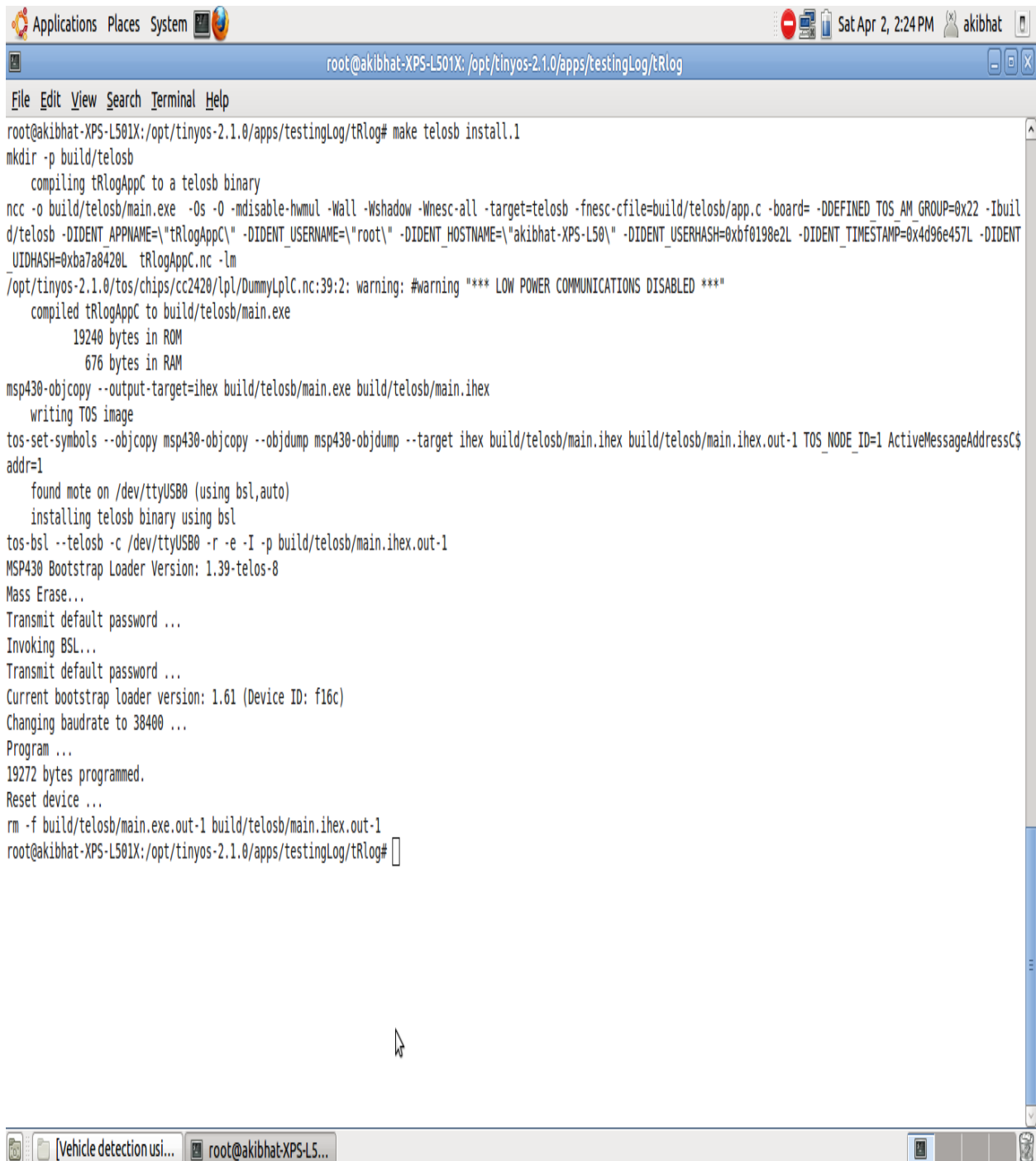
The screenshot shows a terminal window titled "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/tRlog". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The terminal output shows the following commands and their results:

```
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd tRlog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog#
```

The terminal window is part of a desktop environment. The top bar shows "Applications", "Places", and "System" menus, along with system status icons (network, battery, clock) and the user name "akibhat". The bottom bar shows a taskbar with a "Vehicle detection usi..." window and the terminal window itself.

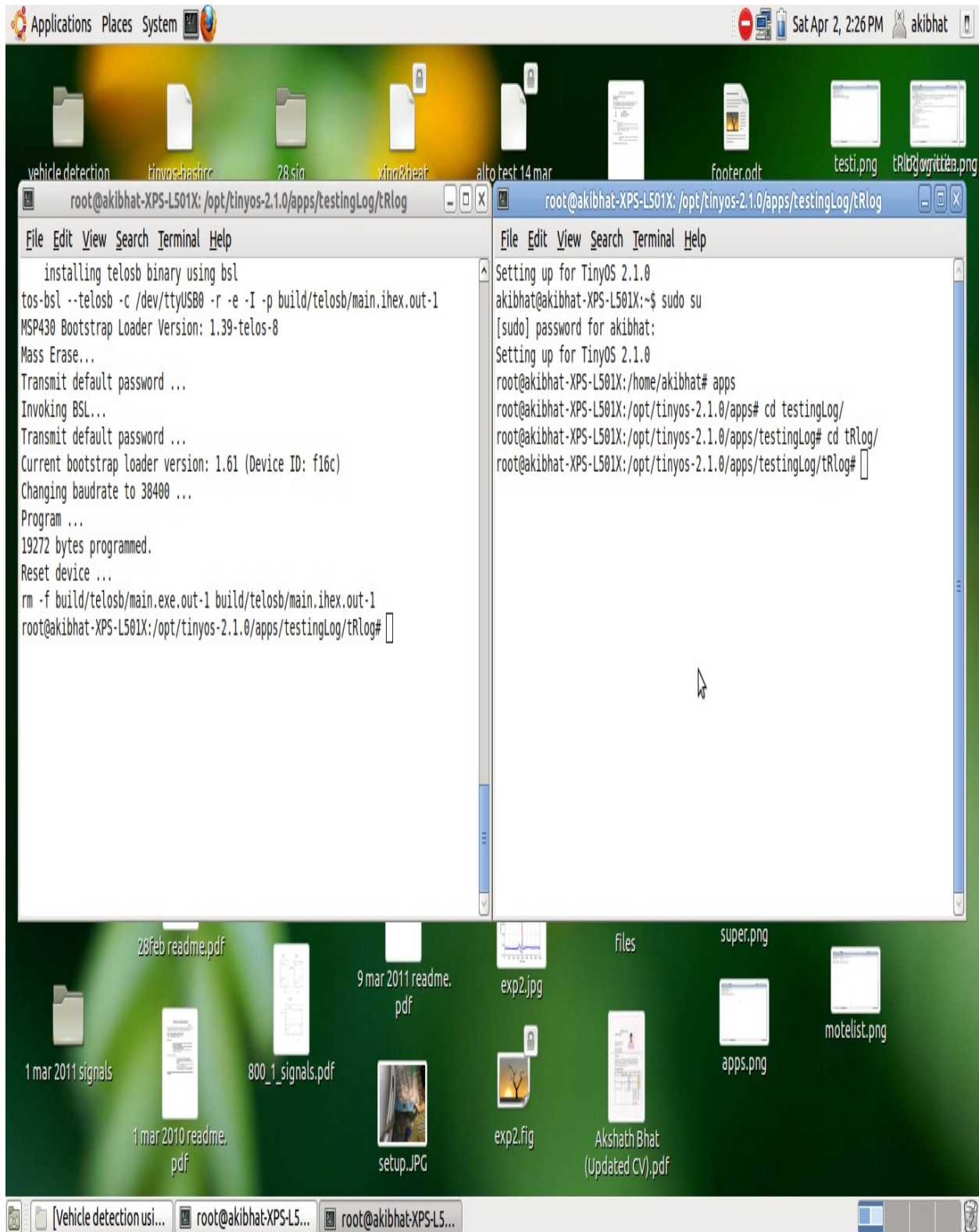
-Type **make telosb install.1** (This is to write the tRlog code into the mote, so that the data collected can be read by the laptop)

Note: Since a new code is written into same mote ie mote B, here the same numbering index as used for SenseLog can be used.



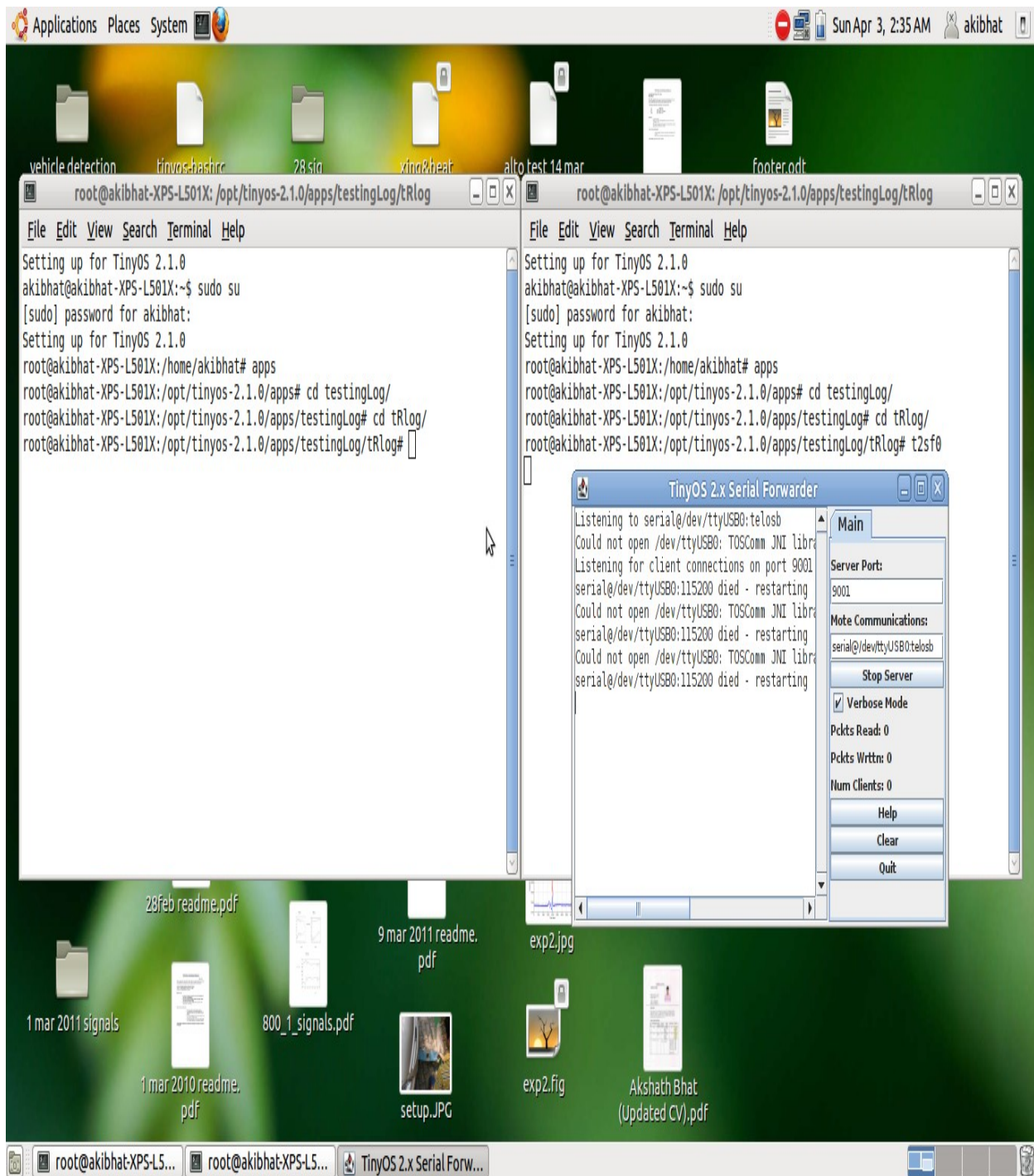
```
root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/tRlog
File Edit View Search Terminal Help
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog# make telosb install.1
mkdir -p build/telosb
  compiling tRlogAppC to a telosb binary
ncc -o build/telosb/main.exe -Os -O -mdisable-hwmul -Wall -Wshadow -Wnesc-all -target=telosb -fnesc-cfile=build/telosb/app.c -board= -DDEFINED_TOS_AM_GROUP=0x22 -Ibuild/telosb -DIDENT_APPNAME=\"tRlogAppC\" -DIDENT_USERNAME=\"root\" -DIDENT_HOSTNAME=\"akibhat-XPS-L501X\" -DIDENT_USERHASH=0xbf0198e2L -DIDENT_TIMESTAMP=0x4d96e457L -DIDENT_UIDHASH=0xba7a8420L tRlogAppC.nc -lm
/opt/tinyos-2.1.0/tos/chips/cc2420/lpl/DummyLplC.nc:39:2: warning: #warning \"*** LOW POWER COMMUNICATIONS DISABLED ***\"
  compiled tRlogAppC to build/telosb/main.exe
      19240 bytes in ROM
      676 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-1 TOS_NODE_ID=1 ActiveMessageAddressC$
addr=1
  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-1
MSP430 Bootstrap Loader Version: 1.39-telos-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 ...
19272 bytes programmed.
Reset device ...
rm -f build/telosb/main.exe.out-1 build/telosb/main.ihex.out-1
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog#
```

Now, retain the above Terminal window and open a new terminal window. Let the two windows be called Left-hand Side window (LHS) and Right-hand Side window (RHS).

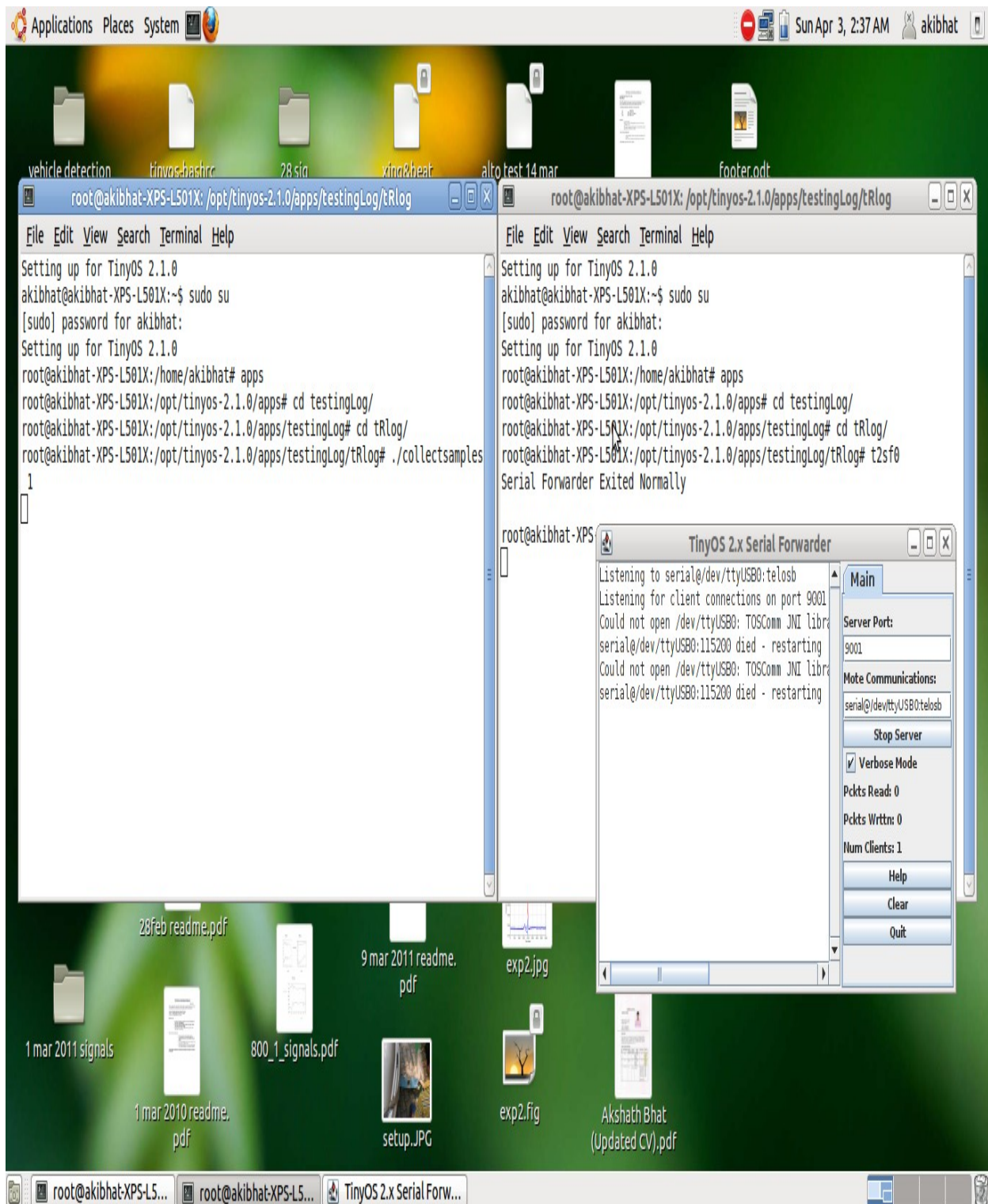


-Type **t2sf0** in the RHS.

This opens the Serial Forwarder of the Tiny OS.



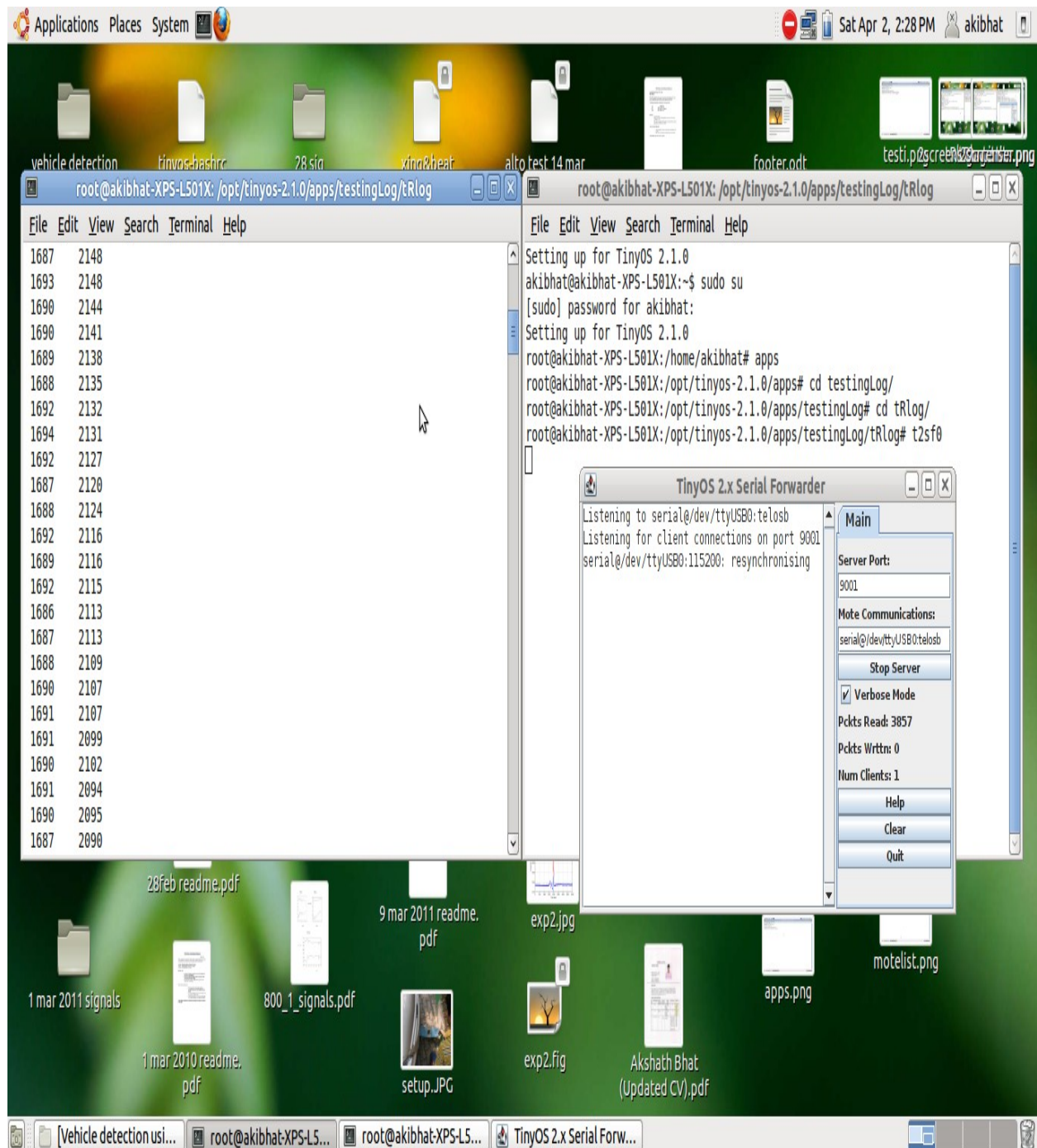
Now, in LHS type **./collectsamples 1** (This is to write the packets of data forwarded by serial forwarder into a text file called 'sdata')



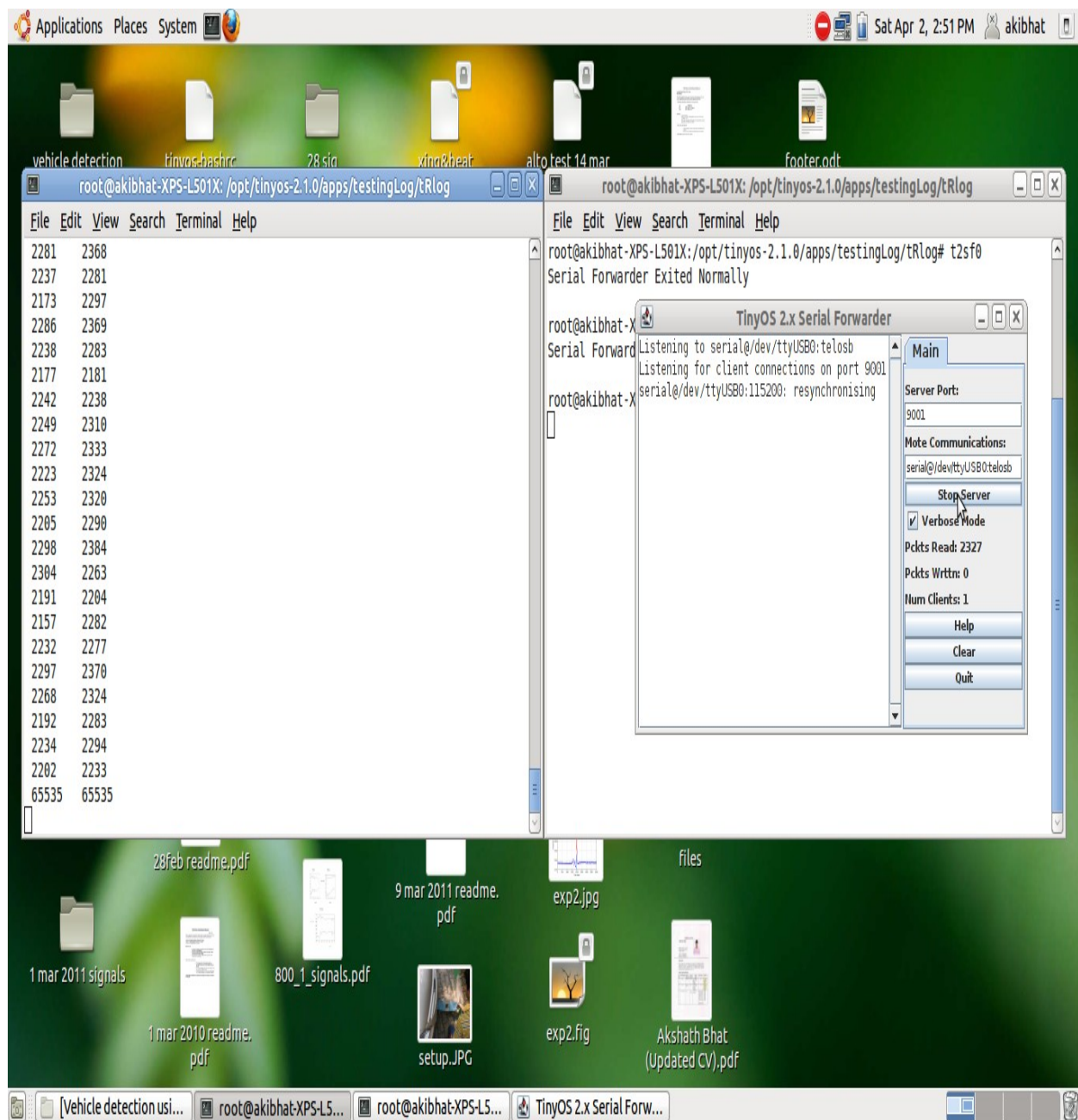
Once the **./collectsamples 1** command is typed the samples are written and the screen is shown below.

The samples are written into a text file '**sdata**' whose path is

opt/tinyos-2.1.0/apps/testingLog/tRlog.

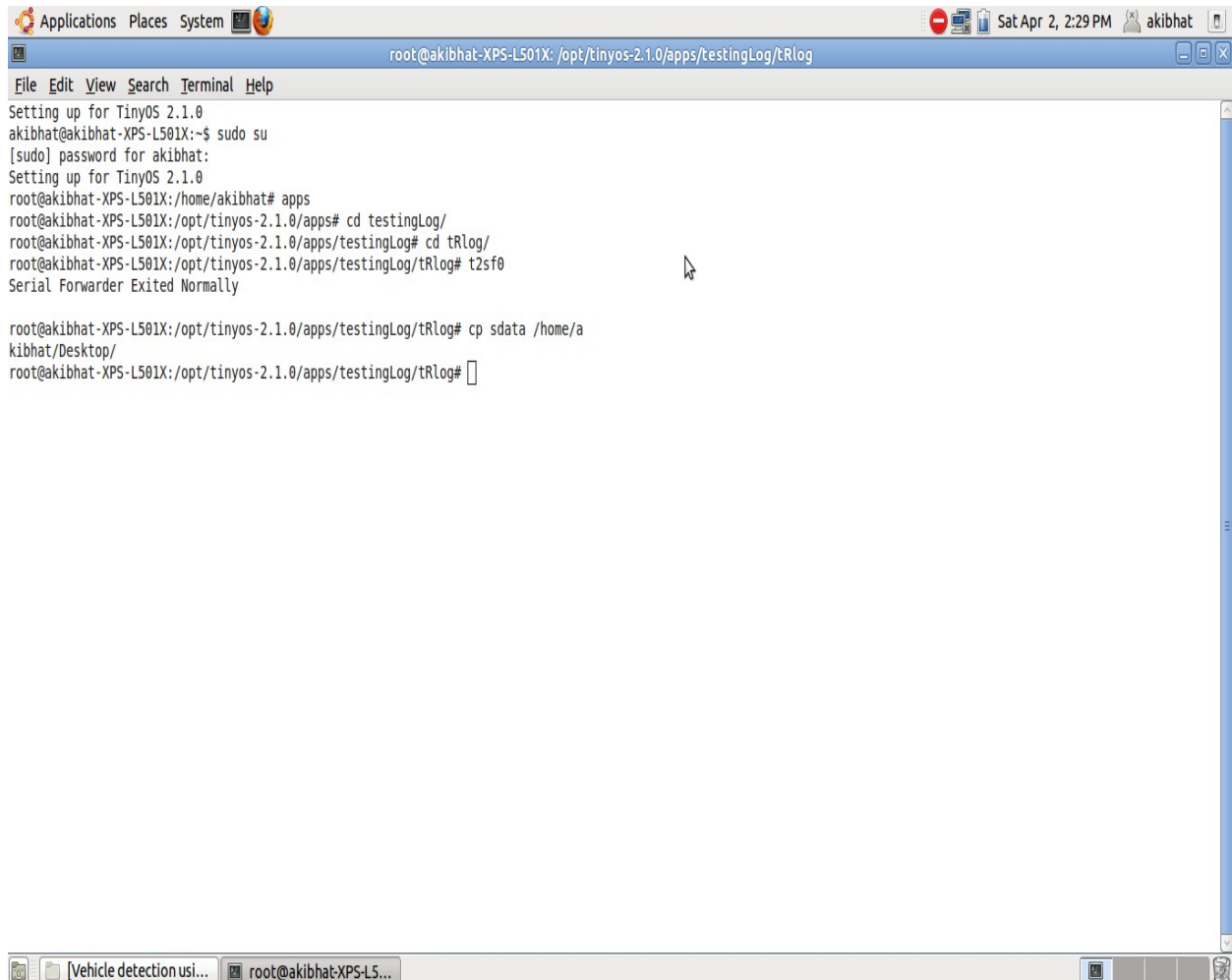


Once writing of data samples stops, stop the serial forwarder by clicking on **Stop Server** button.



Then, close one of the windows after quitting Serial Forwarder.

In the remaining window execute the command **cp sdata /home/akibhat/Desktop/**. This copies the sdata.txt into Desktop. The screen is shown below.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar reads "root@akibhat-XPS-L501X: /opt/tinyos-2.1.0/apps/testingLog/tRlog". The terminal content shows the following sequence of commands and output:

```
File Edit View Search Terminal Help
Setting up for TinyOS 2.1.0
akibhat@akibhat-XPS-L501X:~$ sudo su
[sudo] password for akibhat:
Setting up for TinyOS 2.1.0
root@akibhat-XPS-L501X:/home/akibhat# apps
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps# cd testingLog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog# cd tRlog/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog# t2sf0
Serial Forwarder Exited Normally

root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog# cp sdata /home/a
kibhat/Desktop/
root@akibhat-XPS-L501X:/opt/tinyos-2.1.0/apps/testingLog/tRlog#
```

The desktop taskbar at the bottom shows two open windows: "[Vehicle detection usi..." and "root@akibhat-XPS-L5...". The system clock in the top right corner indicates "Sat Apr 2, 2:29 PM" and the user is "akibhat".

The text file can be renamed and used for further analysis using MATLAB.