Customising Giraff Robot for better User Interaction using Robot Operating System (ROS)

Boga Vishnu Kanth.B,Karthik.S

June 1, 2015

1 Abstract

Giraff Robot used primarily for assisting elderly people by creating Human presence virtually through the video conference provision. A typical Giraff would be as shown in figure 1. It

Figure 1: Giraff Robot

is a differential wheeled robot, where the movement of the robot is provided by two motors independently.It consists of a Computer (32 bit processor) and a low level board (which has ATMEGA 644 microprocessor and motor drivers).

Better User Interaction is done by implementing Accelerometer based Joystick for controlling the Giraff Robot's Speed and Direction ,tracking the proximity of Obstacles using Ultrasonic Sensors and tracking the coordinate or path of Giraff Robot using Encoders which are attached to the Motor Shaft.

1.1 Scope of project

This project does not deal on Video conferencing tools. Its focus is on the functionalities achieved through interaction between On-board Computer and low level board of Giraff Robot

2 Accelerometer based Joystick

Accelerometer based joystick makes maneuvering easy in a disorganised path that is either in a typical household or industrial environment. Accelerometer Sensor would be in the hands of the user. This sensor controls the movement of Giraff by sending (Using ROS) a unique data using to motor drivers for corresponding orientation of the sensor. In this project we used BOSCH Accelerometer BNO055.

2.1 Working Principle of Accelerometer

Accelerometer BNO055 has two components in it. They are
1.Micro Mechanical System
2.Signal processing chip.

2.1.1 Micro Mechanical System

It is a comb like structure as shown in figure 2 below.The blue part in the structure is movable whereas the red part is fixed.This comb structure can be considered as having number

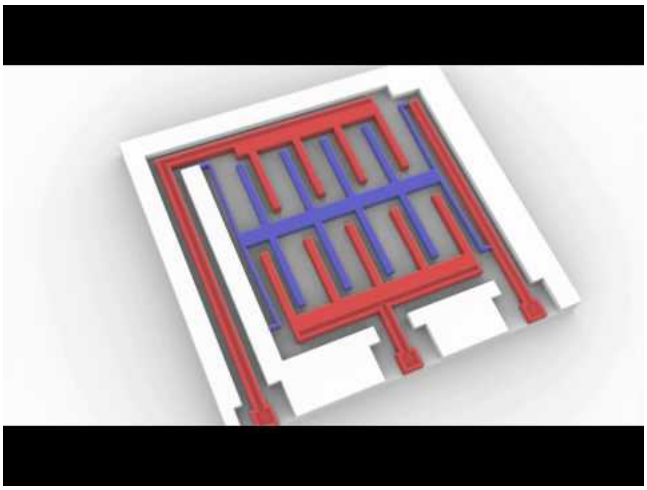


Figure 2: animated view of Micro Mechanical System

of capacitors with red and blue parts as electrode plates.The blue part is very sensitive to movements as it is deflected during acceleration or deceleration of the structure. The gap between each part is being varied during movements.This would result change in the capacitance as the distance between the plates is varied.

2.1.2 Signal Processing Chip

The change in capacitance value is measured by Signal Processing Chip. It performs signal processing techniques like filtering and converts change in capacitance into an output signal with good precision and accuracy. In BNO055, signal processing chip output is capable of producing the functionalities of Accelerometer, Magnetometer, Gyroscope.Using Data fusion you can obtain all above functionalities and communicate the output to the Host processor.In this project host Processor used is **Arduino UNO**. The Communication between UNO and BNO055 signal processing chip is done through I^2C communication.

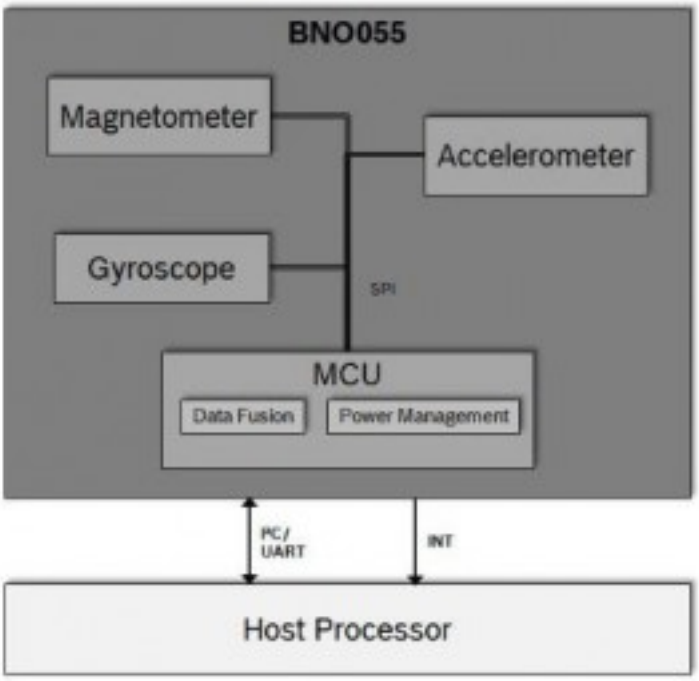


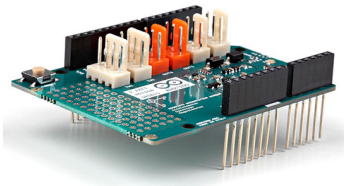
Figure 3: Architecture of signal processing chip BNO055

2.2 Interfacing BNO055 with UNO

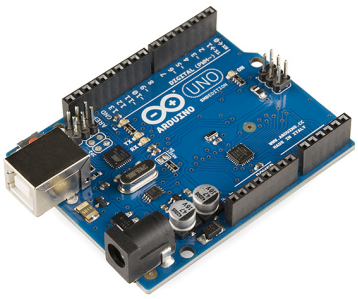
Bosch Sensortec GmbH provide libraries like NAxisMotion.h which contains the bridge code between the BNO055 and the Arduino Environment. So using these we can obtain the output of BNO055 sensor to the UNO registers or peripherals . So, In arduino IDE we can actually manipulate this data according to our application(controlling the motor drivers based on orientation of sensor in user's hand). How manipulation is done can be studied in Interpolation Method.

2.2.1 Implementation

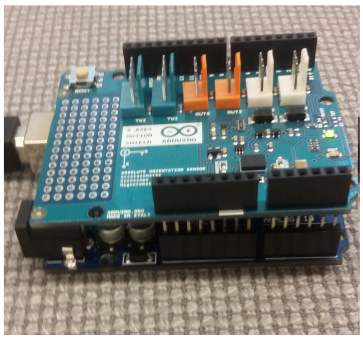
We have used 9 Axes motion shield which has BNO055 embedded on it. 9 Axes motion shield can be easily assembled on UNO.



(a) 9 Axes motion shield



(b) Arduino UNO



(c) Assembled and in Datum Orientation

Figure 4: 9 Axes motion shield can be assembled on UNO

Accelerometer functionality facilities better control of Giraff. In UNO we had put configuration mode into ACCONLY. In this mode the other sensors (magnetometer,gyro) are suspended to lower the power consumption and BNO055 behaves like a stand-alone acceleration sensor.For every orientation it has a unique X,Y,Z coordinates.Variation of X,Y,Z coordinates based on the orientation is shown in figure 5. We had calibrated and ruled out Z

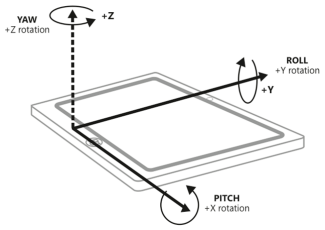


Figure 5: Variation of Coordinates based on Rotation

coordinate because X,Y coordinates alone is sufficient for controlling Giraff Robot. By using interpolation method we had mapped the X,Y coordinates (output from accelerometer that is available on Arduino registers) to corresponding PWM output for the motor drivers.

Interpolation Method For our convenience we made accelerometer orientation as shown in figure 4c which is flat to the ground surface as the datum orientation because the X,Y coordinates (output from accelerometer which we had in the UNO registers) is 0. In this datum orientation of accelerometer, the Giraff robot should not move in any direction.As we need the Giraff Robot to move Forward,Backward,Right,Left directions, we utilised four orientations of the sensors and based on the angle of inclination in each orientation the PWM output to the motor driver is interpolated as follows.

2.2.2 Forward Orientation

In this orientation as shown in figure 6 the Y-coordinate is 0 and X-coordinate will increase from 0 to 9.5 based on the inclination of sensor from the datum orientation as shown in figure 4c. This value of X-coordinate is manipulated in the equation below so that we can

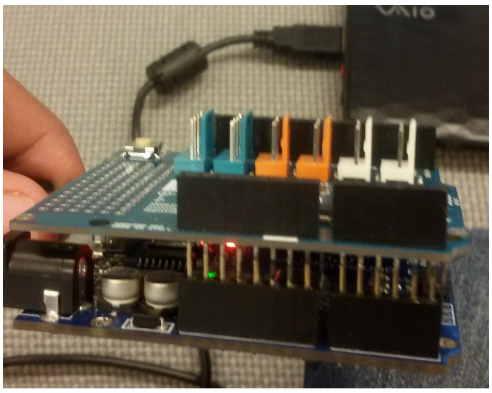


Figure 6: forward orientation

obtain PWM output which can be fed to motor drivers.

$$PWMx1 = 14 * x - 63;$$

Here the 'x' register stores the X-coordinates. Considering the ergonomics, only the X-coordinates greater than 4.5 are fed to the equation. So ,If the X-coordinates increases by increasing the inclination of the sensor from 4.5 to 9.5 the PWM output('PWMx1' register) value changes from 0 to 70. How this PWM value effects the motor speed is discussed further in new section under Motor Control.How this PWM value is reached to motor driver is discussed under section Interfacing Accelerometer and motor driver using Robotic Operating System(ROS tool).

2.2.3 Backward Orientation

In this orientation as shown in figure 7 the Y-coordinate is 0 and X-coordinate will decrease from 0 to -9 based on the inclination of sensor from the datum orientation as shown in figure 4c. Similarly the PWM output('PWMx2' register) equation would change from 0 to

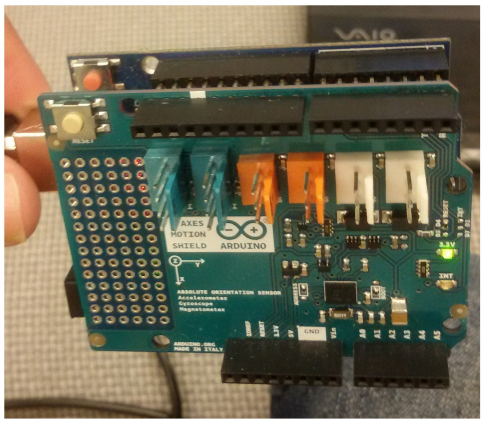


Figure 7: Backward orientation

-70 for change in X-coordinate from -4.5 to -9.5

$$PWMx2 = 14 * x + 63;$$

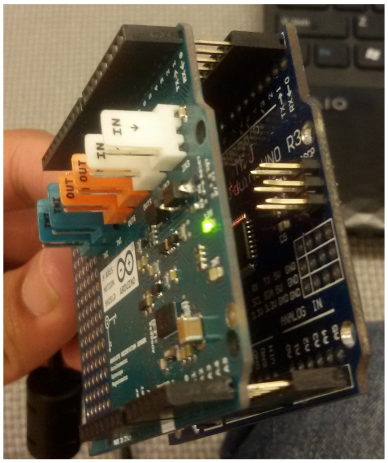


Figure 8: left orientation

2.2.4 Left Orientation

In this orientation as shown in figure 8 the X- coordinate is 0 and Y-coordinate will increase from 0 to 9 based on the inclination of sensor from the datum orientation as shown in figure 4c.

Similarly the PWM output('PWMy1' register) equation would change from 0 to 70 for change in X-coordinate from 4.5 to 9.5

$$PWMy1 = 14 * y - 63;$$

2.2.5 Right Orientation

In this orientation as shown in figure 9 the X-coordinate is 0 and Y-coordinate will decrease from 0 to -9 based on the inclination of sensor from the datum orientation as shown in figure 4c. Similarly the PWM output('PWMy2' register) equation would change from 0 to

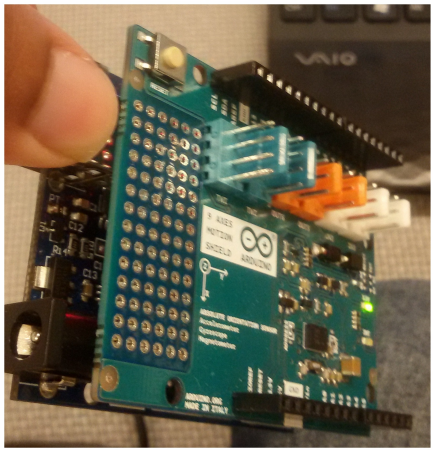


Figure 9: Right orientation

-70 for change in X-coordinate from -4.5 to -9.5

$$PWMy2 = 14 * y + 63;$$

3 Deploying Ultrasonic Sensors

3.1 What is Ultrasonic Sensor?

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats or dolphins do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1" to 13 feet. It's operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

The basic principle of work: (1) Using IO trigger for at least 10us high level signal, (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back. (3) If the signal back, through high level, time of high output duration is the time from sending ping till it returns. Test distance = high level $time * velocity$ of sound $(340M/S)/2$.

3.2 Features

- Power Supply :$+5V$ DC
- Quiescent Current : $<$ 2mA
- Working Current: 15mA
- Effectual Angle: $<$ 15 deg
- Ranging Distance : $2cm$ - $400cm$/1" - $13ft$
- Resolution : $0.3cm$
- Measuring Angle: 30 deg
- Trigger Input Pulse width: $10uS$
- Dimension: $45mm * 20mm * 15mm$

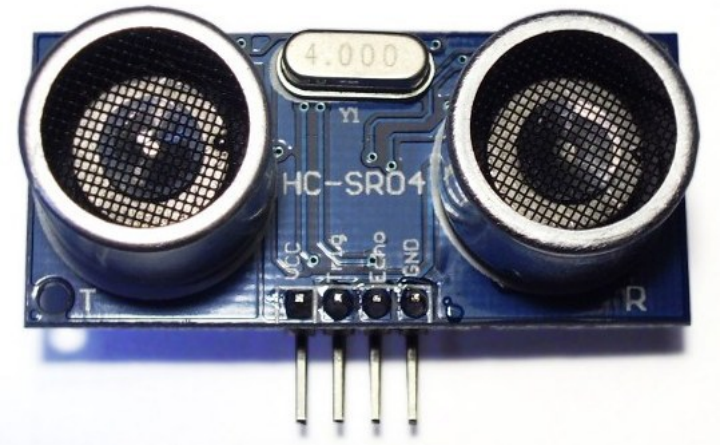


Figure 10: UltraSonic Sensor

3.3 Timing Diagram

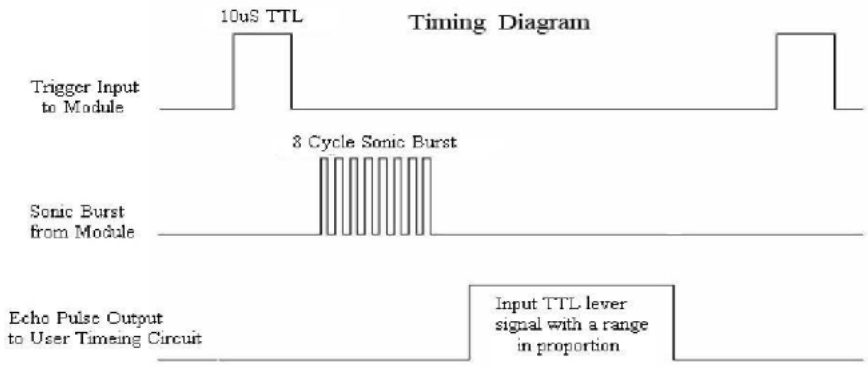


Figure 11: Timing Diagram

The Timing diagram is shown above. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion. You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula:$uS/58 = centimeters$ or $uS/148 = inch$; or: the range = $\left[\frac{highleveltime*velocity(340m/s)}{2}\right]$.

Explanation:

The speed of the Ping is 340m/s.

Therefore, to cover 1 meter, the signal takes = 1/340 seconds

=0.002941 s/m

=0.002941/100 s/cm

=0.00002941 s/cm

=29.41 uS/cm

≈ 29 uS/cm.

The ping travels a distance which equals two times the distance of obstacle from the sensor. Therefore, the distance of obstacle from the sensor is $= \frac{\frac{timetakenbythesensor}{29}}{2}$ $= \frac{timetakenbythesensor}{58}$.

The same applies for inches calculation.

3.4 Hardware Interface

In the figure, the Ultrasonic Sensors(2 at the front of the Giraff bot and 1 at the back)are integrated with Arduino Mega 2560(placed at the top of the bot). There are 4 pins in the HC-SR04 Ultrasonic Sensor Module. They are 5V Supply(Vcc), Trigger pin, Echo pin and 0V Ground(Gnd). Using Microcontroller and Arduino IDE, you can program the code for Ultrasonic Sensor. I have used Arduino Mega 2560 as my microcontroller and Arduino IDE for programming the code. You can download Arduino IDE software from `http://www.arduino.cc/en/Main/Software`. The Arduino Mega 2560 is a microcontroller

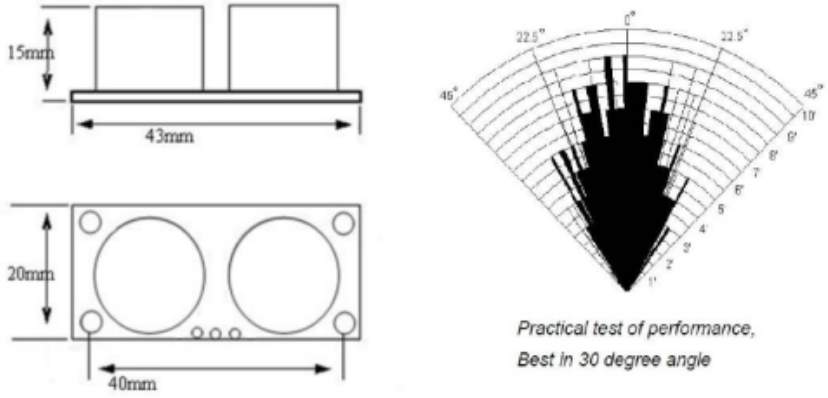


Figure 12: Performance level

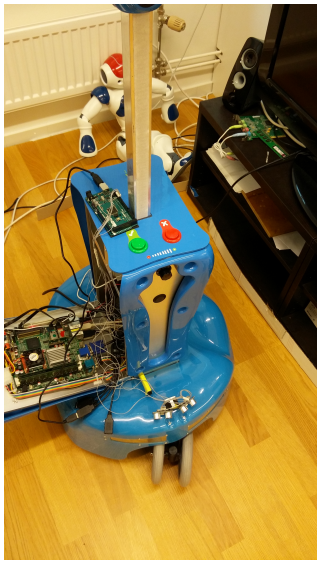


Figure 13: Hardware interfacing diagram

board based on the ATmega2560(please refer to the datasheet for more details on ATmega2560). It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs(hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

4 Motor Control and Encoders

Giraff Robot consists of on-board Computer and Micro Controller board . Micro Controller board consists of ATmega644 processor and Motor Drivers **A3959** and many small components for other functionalities. Encoders are fixed to motors shaft.ATmega644 peripherals are connected to Motor Driver (A3959) pins and Encoder **AMT102** Channels.

Arduino IDE is customized so that ATmega644 can function in Arduino Environment. This is done by re-writing the core libraries and headers and placing them into new directories within the Arduino environment directories. For more information about Customizing Arduino IDE you see in this link : http://playground.arduino.cc/Main/CustomizeArduinoIDE

4.1 Interfacing Motor Driver A3959 with ATmega644

A3959 Motor Driver is chosen because it can output Pulse Width Modulated(PWM) current control for DC Motor . Its Operating Range is 50 Volts and is capable of output Currents from -3 to +3 Ampere. **PHASE** terminal is used to control the direction of Motor. **ENABLE** terminal is used to control the Speed of Motor by applying PWM signal to it.**OUTA** and **OUTB** terminals are connected to Motor. All other connections are made based on the suggestions in Data Sheet of A3959.

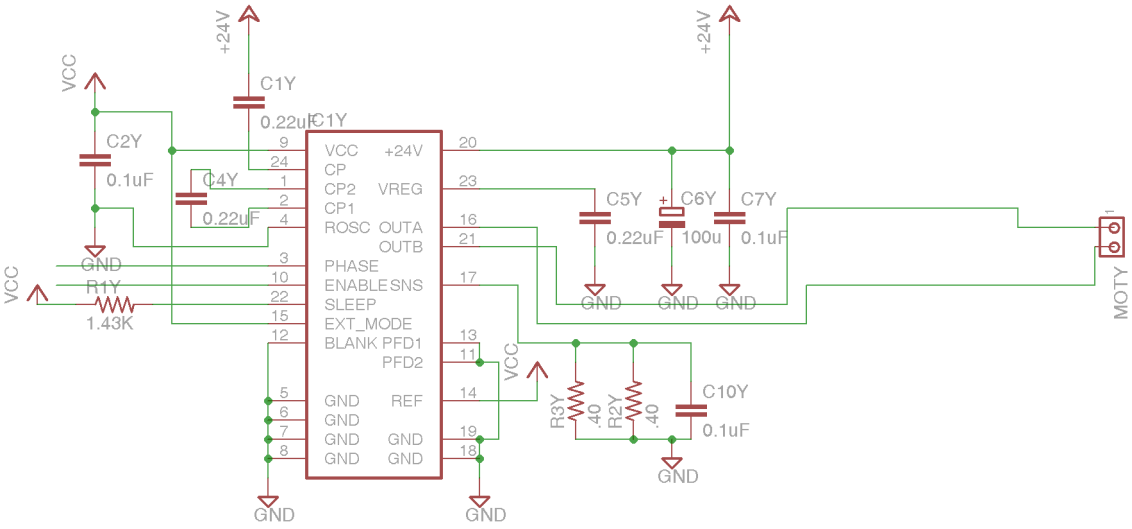


Figure 14: Schematics of A3959 in Micro controller board

As there are two Motors for movement of Giraff Robot, two Motor Drivers A3959 are used. Enable and Phase terminals of the Motor Drivers are connected to the ATmega644 output peripherals. As ATmega644 can work in Arduino Environment, we can write a code in .ino file and able to control the motors by controlling the PHASE and ENABLE pins which are connected to ATmega644.

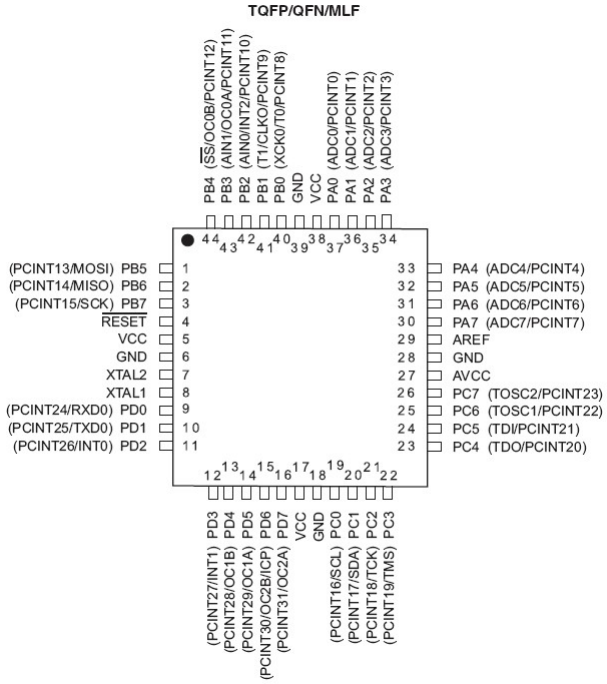


Figure 15: ATmega644 PIN diagram

4.1.1 Controlling the speed of Motors

As PD4 and PD6 pins of ATmega644 which are capable of PWM output are connected to two Enable Pins of A3959 Drivers. So, by varying the decimal value sent to PD4 and PD6 from 0 to 255 we can obtain EMF range of 0-24 Volts across the OUTA and OUTB terminals of A3959 which are further connected to the Motor terminals.

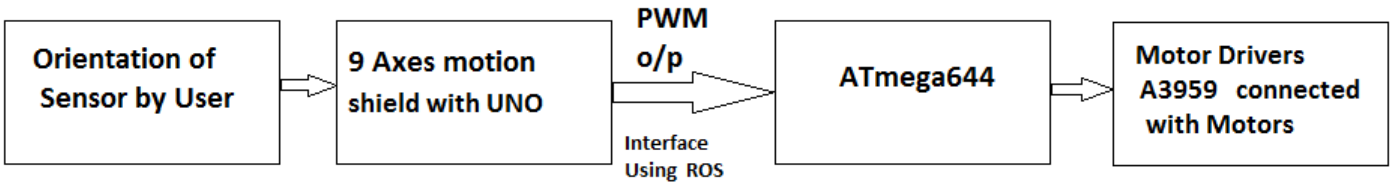
4.1.2 Controlling the direction of Motors

PHASE	OUT_A	OUT_B
0	Low	High
1	High	Low

Figure 16

As discussed earlier, using Phase pin of A3959 we can change the direction of motors. The table shown in figure 16 gives an glimpse of how Phase pin effects the OUTA and OUTB which are connected to motors. PB4 and PC6 Digital output pins of ATmega644 are connected to Phase pins of A3959 . So, by changing PB4 and PC6 from high to low or vice versa , we can change the direction of EMF applied across the Motor terminals.

As we know from section 1.2 that Accelerometer sensor would Output PWM based on the Orientation of the Sensor. Now this PWM is received by the ATmega644 , where further manipulations are done based on the input received and Phase, Enable Pins are controlled



so that the motors direction and speed are controlled.
How the PWM value is transferred Accelerometer Module and Motor Control Module is discussed in further section under Interface Using Robot Operating System(ROS).

4.2 Odometry Using Encoders AMT102

Motor Encoders are used in order to track the parameters like the Robot's displacement,Velocity or its coordinate in a path etc. In Giraff Robot, the motor encoders are Capacitive type AMT102 which are assembled to the rotor shaft of motors as shown in figure 17.

Figure 17: Encoder attached to one of the Giraff Robot Wheel

4.2.1 Working Principle and its Operation

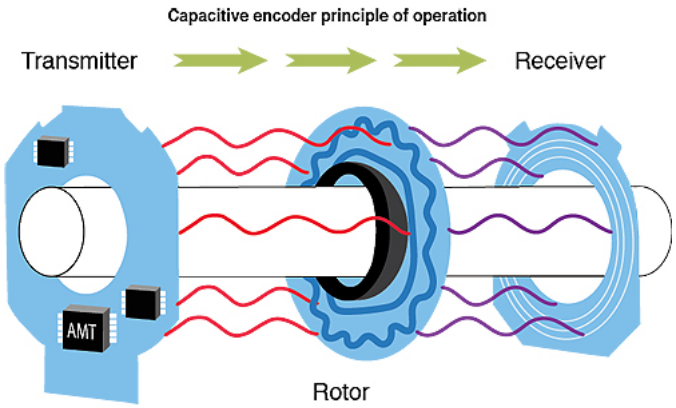
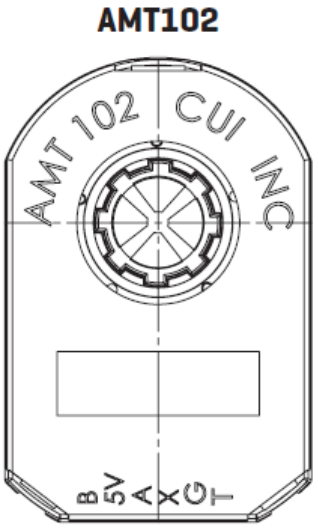


Figure 18: Animated view

Capacitive encoders use a high-frequency signal in the range of around 10 MHz, which gets beamed from a transmitter to a receiver through a capacitive rotor disk. The metal pattern on the rotor amplitude modulates the signal in a repetitive and predictable way. The resulting modulated signal is detected at the receiver and demodulated. An Application

Specific Integrated Circuitry(ASIC) converts the demodulated information to pulses that indicate the position of the motor shaft. Capacitive Encoders AMT102 has 6 PINS as shown in figure 19.This is a three channel Encoder where we can find the incremental position as well as absolute position of the motor. A,B Channels of AMT102 are used for obtaining the incremental position and produces Quadrature signals in the form of square waves. INDEX Channel X would give a pulse for every 360 degree rotation of the motor shaft.So,using X channel we can obtain absolute position.



#	AMT102
B	B CHANNEL
5V	+5 V
A	A CHANNEL
X	INDEX CHANNEL
G	GND
T	UNUSED

Figure 19: AMT102 PIN OUT

Decoding the Quadrature Signals The Quadrature Signals of A,B Channels would be 90 degrees out of phase as shown in figure 20

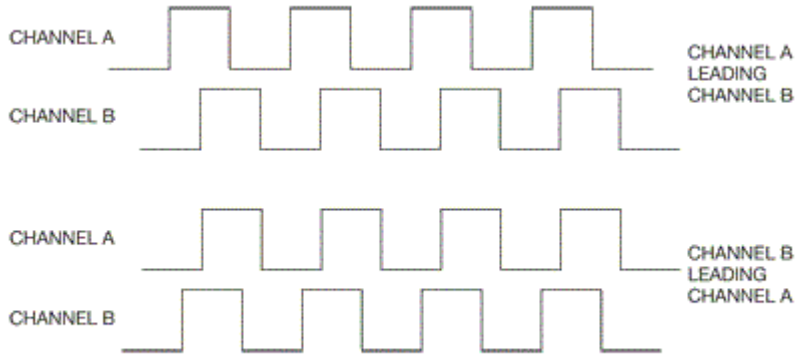


Figure 20: Quadrature Signals of Encoder

If the motor is rotating Clockwise **(CW)** then Channel A Leads Channel B. If the motor rotates Counter Clockwise **(CCW)** then Channel B leads Channel A. By monitoring both the number of pulses and the relative phase of signals A and B, you can track both the position and direction of rotation. So, we have used only A,B channels to reduce complexity and are connected to ATmega644.

4.2.2 Interfacing Encoders with ATmega644

As there are two motors, two encoders are used and named them as left and right encoders as they are attached to left and right motors of Giraff Robot respectively. Channel A of left ,right encoders are connected to PD2,PD3 pins of ATmega644. From the pin diagram of ATmega644 shown in figure 14 Interrupts INT0 and INT1 can be used in PD2,PD3 respectively.Channel B of left and right encoders are connected to PC7,PC0 respectively.We know that ATmega644 can be used in Arduino Environment.Counters are initialised using arduino registers

attachInterrupt(0,signalCheck,RISING)

The above command in Arduino IDE initialises the interrupt 0 which works on PD2(Channel A of left encoder) and invokes Interrupt Service Routine **(ISR)** named signalCheck when ever it observes Rising Edge of Channel A.

Now in ISR by comparing the position of channel B at rising edge of channel A, we can decode direction of the motor shaft rotation, as shown in figure 20 , counter register is **incremented** for Clockwise **CW** rotation or **decremented** for Counter Clockwise rotation **CCW**.

```
signalCheck()

{

if (digitalRead(PC7))
    count++;
    else
    count–;

}
```

Resulting Counter register has number of **Wheel ticks**.Using Wheel ticks along with timer registers we can calculate parameters like Robot's displacement, velocity ,current coordinates in a path etc.

5 Integrating modules Using ROS

In previous discussions we learnt different modules like, how Accelerometer outputs PWM, how PWM signals are used to control the Giraff robot , how Odometry can be done using Encoders. In this section we are going to learn Using **Robot Operating System** (ROS) , clubbing all modules so that Giraff Robot can be controlled effectively.

Introduction to ROS : ROS (Robot Operating System) is a BSD-licensed system for controlling robotic components from a PC. A ROS system is comprised of a number of independent nodes, each of which communicates with the other nodes using a publish/subscribe messaging model.Depending on how the system is configured, any node may need to communicate with any other node, at any time. The primary mechanism for ROS nodes to exchange data is to send and receive messages. Messages are transmitted on a topic and each topic has a unique name in the ROS network. If a node wants to share information, it will use a publisher to send data to a topic. A node that wants to receive that information will use a subscriber to that same topic. Besides its unique name, each topic also has a *message type*,which determines the types of messages that are allowed to be transmitted.

This publisher/subscriber communication has the following characteristics:

- Topics are used for many-to-many communication. Many publishers can send messages to the same topic and many subscribers can receive them.
- Publisher and subscribers are decoupled through topics and can be created and destroyed in any order. A message can be published to a topic even if there are no active subscribers.

The concept of topics, publishers, and subscribers is illustrated in the following image: For

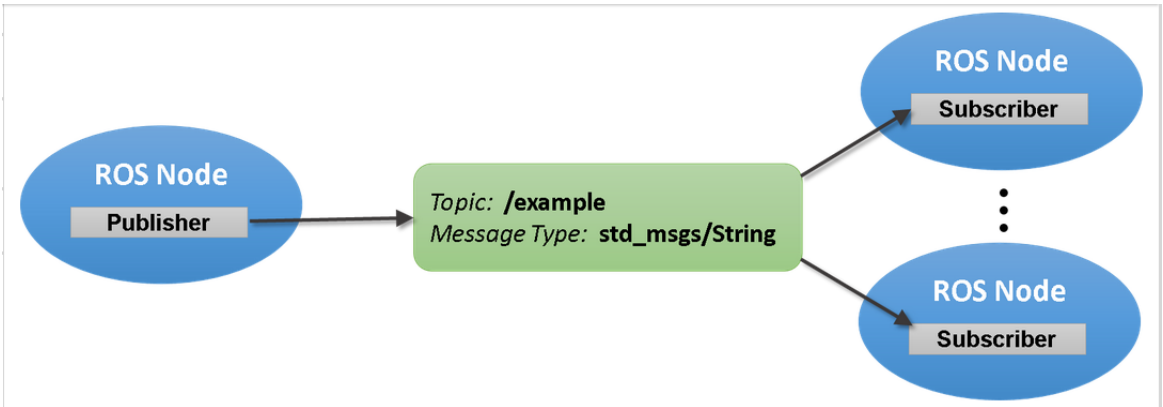


Figure 21

further features and uses of ROS you can see in this link: http://wiki.ros.org/ROS/Tutorials

We had installed ROS indigo in Giraff's Computer which is on board Computer. ROS indigo installed in laptop where Accelerometer is connected and also we can can get Giraff Odometry in that laptop. ROS libraries has to be Setup in Arduino IDE.

5.1 Interfacing ROS with Arduino

Once ROS libraries was Setup in Arduino home directories, we can use a package called **rosserial** which is used for ROS communication protocol that works over your Arduino's UART. It allows your Arduino to be a full fledged ROS node which can directly publish and subscribe to ROS messages, publish TF transforms, and get the ROS system time. This package has a node **serialnode.py** that initialises baud rate and COM port to which Arduino is connected.

5.1.1 Accelerometer Node

Accelerometer Node is uploaded on UNO which has 9 Axes motion shield assembled on it.Accelerometer node uses Int8 message in stdmsgs package.This node publishes two topics namely **leftWheel** and **rightWheel**. These topics consists of PWM data, generated in accelerometer, based on the Orientation and inclination of the Sensor that is discussed under Section 2.2 .For complete reference of Accelerometer node you can see in this link :https://github.com/bogamct/Giraff-ROS-Nodes/blob/master/Acceleromoter-NODE.

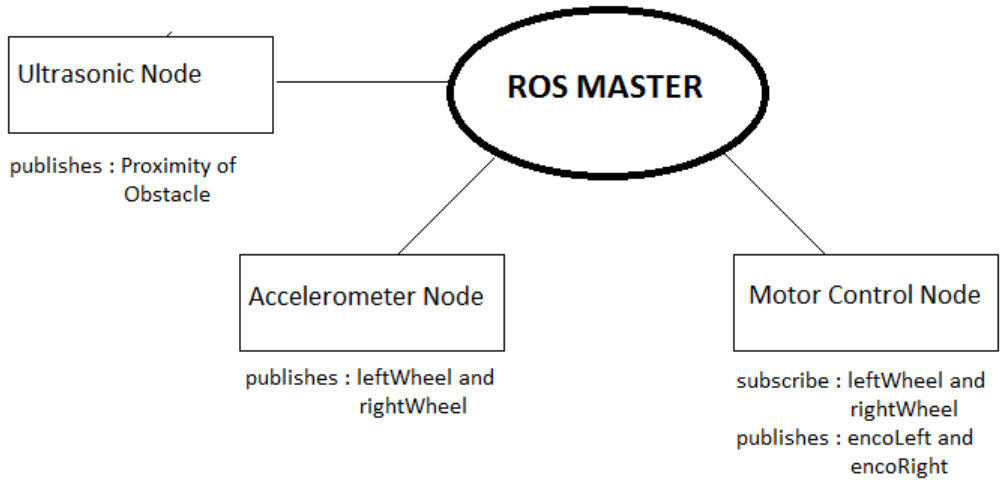
5.1.2 Ultrasonic Node

Ultrasonic Sensor node is uploaded to Arduino Mega 2560. This node publishes to a topic called chatter. Another node can subscribe to the topic chatter and use the data of ultrasonic sensor. The code which i have developed while working with multiple Ultrasonic Sensors can be fetched from https://github.com/Karthik-Sur under the Ultrasonic_Sensor Repository-Timer_ROS - Timer_ROS.ino. The code has comments on each line which explains the code. ROS comes to the rescue when we need to connect two or more programs to each other.To this, you can further code the program to make the robot move without hitting the obstacles by receiving data from the sensors and using that data to control movements of the Giraff robot.

5.1.3 Motor Control Node

Motor Control Node is uploaded from Arduino IDE ,to ATmega644(Micro controller board) on the Giraff Robot. This Node subscribes **leftWheel** and **rightWheel** topics which being published by Accelerometer Node. By this we can obtain the PWM data on Arduino registers(since ATmega644 can work in arduino Environment) where further manipulations ,like the decision of PHASE pin of Motor Drivers based on sign of PWM signal and the absolute value of PWM signal is assigned to ENABLE pin of Motor Driver .So,finally we can observe the variation of, speed of motor based on inclination at each orientation of Accelerometer Sensor.

Further, Motor Control Node uses Int16 message in stdmsgs package and publishes two topics namely **encoLeft** and **encoRight** which has number of Wheel ticks info (how it is generated is discussed in section 5.2.2), of left and right Wheels of Giraff Robot respectively.For complete reference of motor Control node you can see in this link :https://github.com/bogamct/Giraff-ROS-Nodes/blob/master/MotorControl-encoders-ROSnode.



5.1.4 Interaction between Nodes

As Motor control Node is at on-board computer and Accelerometer Node is on the laptop where Accelerometer is connected which is controlled by User. In order to interact nodes in Multiple Machines we need to establish ROS Network Setup and we need to select any of one computer to start ROS master. All nodes must be configured to use the same master, via **ROS_MASTER_URI**. . Once this is done you can run each node and the ROS master executes all nodes.

In our project, we made Giraff on-board computer as MASTER and configured its ROS_MASTER_URI in all systems. We tracked the PWM data and Encoder Wheel ticks of both Left Wheel and Right Wheel of Giraff Robot using command **rostopic echo** topicName in both the computers. In this way we have created a User interface in the laptop where Accelerometer is connected and able to get path of Giraff robot traversed , and its velocity at every instant. We are able to control the speed and direction of Giraff Robot based on the Inclination and Orientation of Accelerometer Sensor respectively.

6 Bibliography

1. http://wiki.ros.org/ROS/Tutorials/MultipleMachines
2. http://forum.arduino.cc/index.php?topic=106043.0
3. http://www.micropik.com/PDF/HCSR04.pdf
4. https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit?pli=
5. http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf.
6. http://www.arduino.org/products/arduino-9-axes
7. http://www.cui.com/product/resource/amt100-series-modular-encoders.pdf
8. http://playground.arduino.cc/Main/RotaryEncoder