# COMPUTER SCIENCE

PROJECT REPORT
C++ GUI – SEMESTER II


UNIVERSITY OF BURGUNDY

Prof. Yohan FOUGEROLLE

Cansen JIANG

Vishnu VEILUMUTHU

VIBOT – BsCV

# ABSTRACT

The project is based on design of robotic framework with C++ as programming language. The application contains a server and client network connecting a robot and workstation. The server sends data to the connected clients like sensor parameters. The server section manages password for security access, camera display and serial communication for sensors. The client displays and monitors the data from the other end.

# TABLE OF CONTENTS

# LIST OF FIGURES

## INTRODUCTION

The v.m.v.i.s. is a C++ GUI software application designed for TCP/IP Communication, Camera display, Sensor 3D display and Serial terminal. The application is a simple robotic framework design which communicates between server robot and a client workstation. The server robot monitors the parameters like sensors and camera then sends the data to workstation for processing.

### Features

- The server is started with various levels of access functions.
- The client is connected only if the password is valid, which increase the security of data.
- The camera is started with safety mode, which saves the program from crash when accessing other data.
- The 3D model of data can be viewed both in client and server.
- The sensor data can be viewed in terminal and also can be printed.

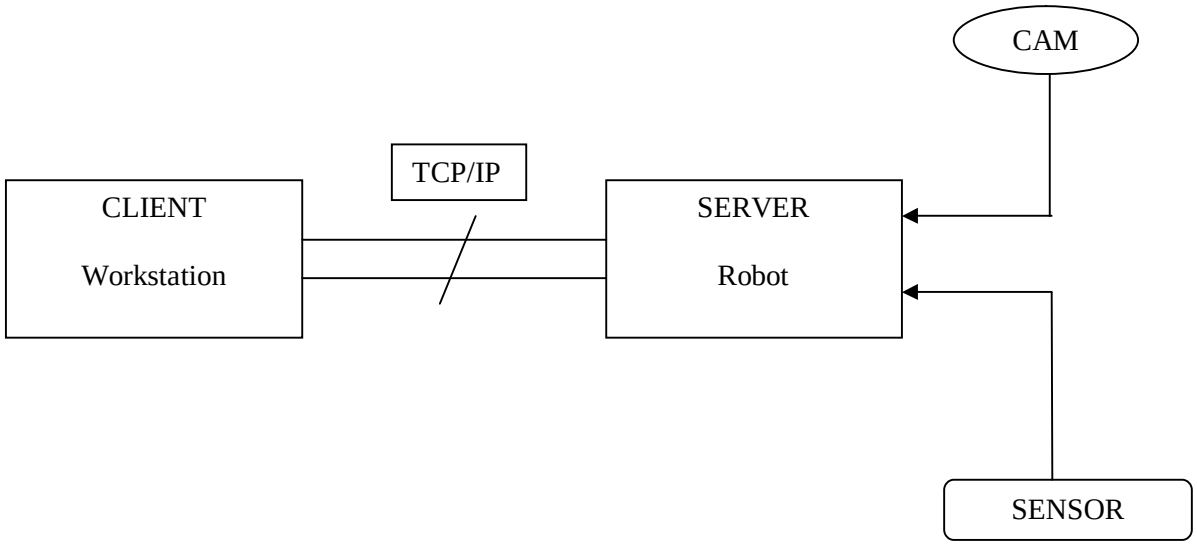


Figure 1: Data flow model

## CHAPTER 1

## 1 BOOT SECTION

The application is started with the display of the boot dialog box. The boot dialog acts as a welcome screen to choose between server and client page. The dialog box contains two pushbuttons, which opens server or client page and closes the boot dialog box.

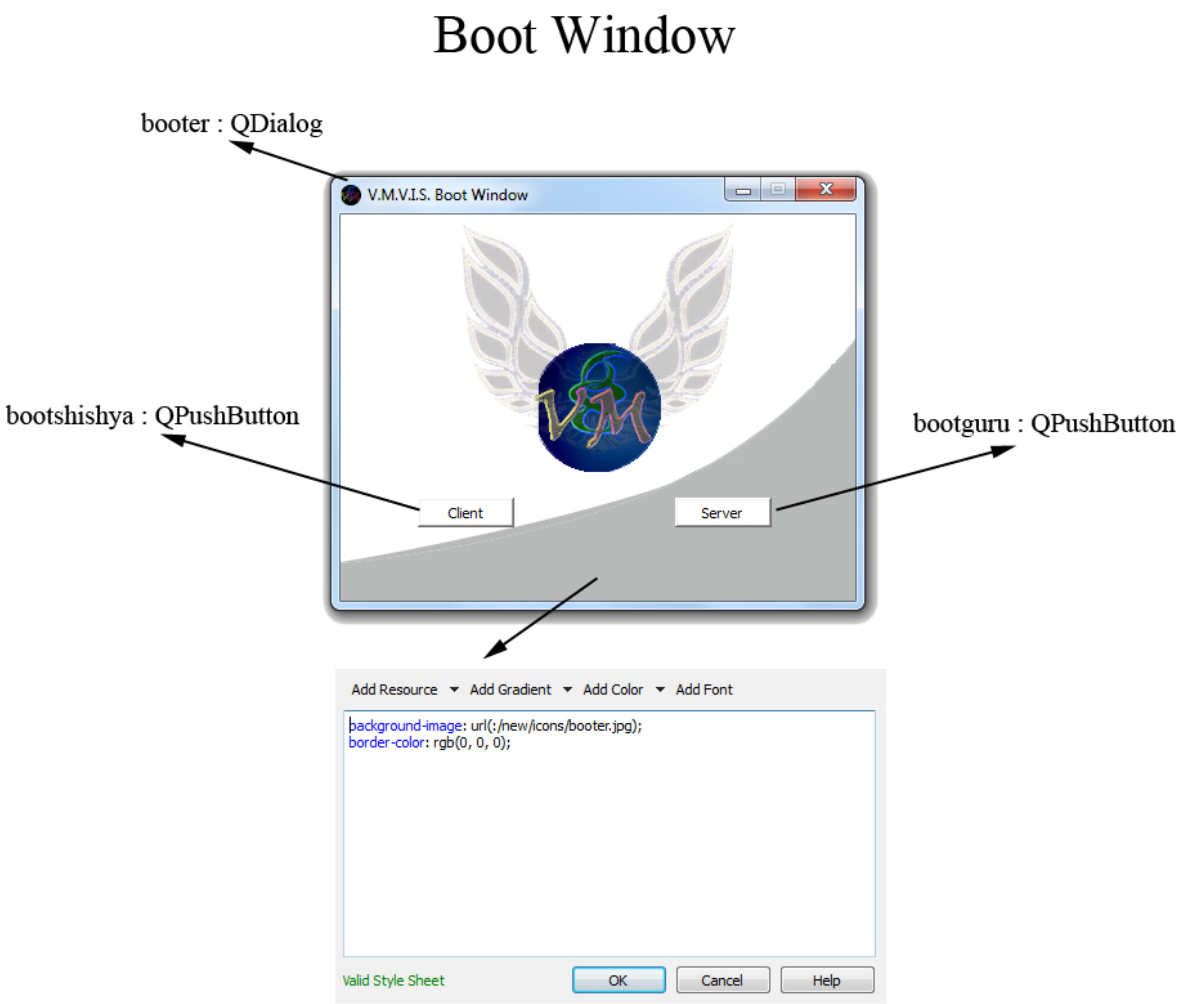


Figure 2: Boot Window

### 1.1 Project Tree

Pro file – The Pro file of the project contains all the necessary instructions to add for the project.

Main class – The main class is designed very simple to display the boot dialogbox when the program is started.

Resource files – The resource file contains all the necessary images and icons for the project.

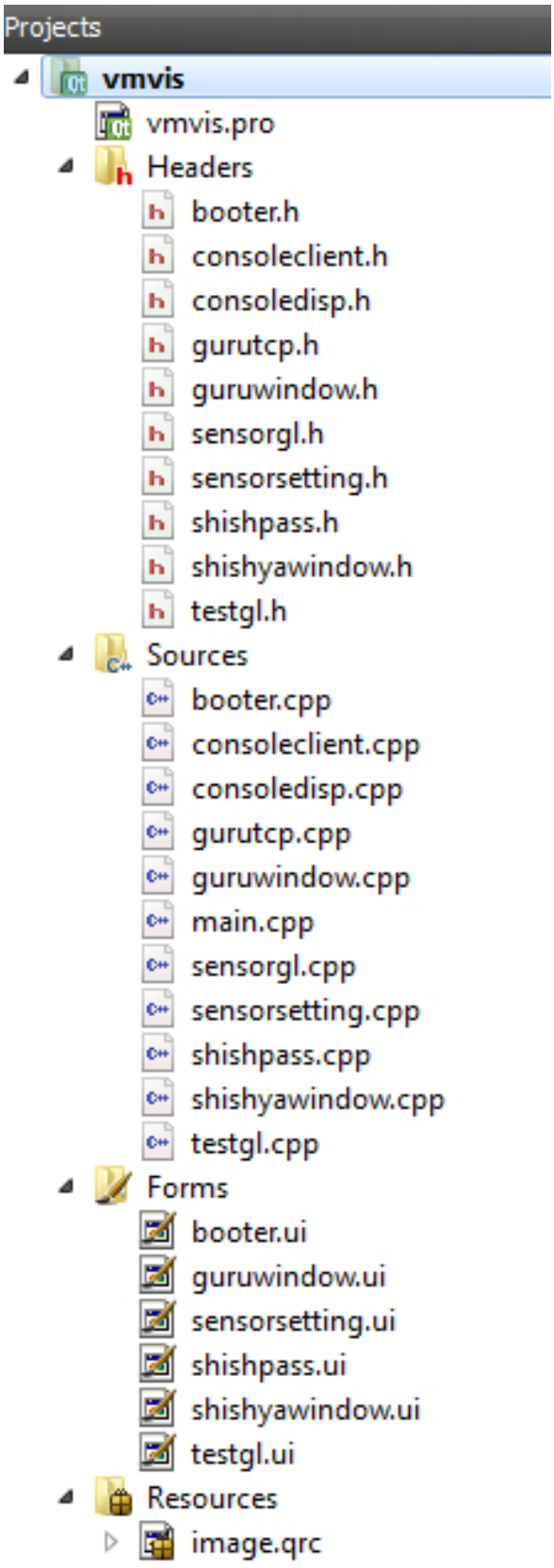


Figure 3: Project Tree diagram

# CHAPTER 2

## 2 NETWORK SECTION TCP/IP

The network section contains a simple TCP/IP communication between a server and client where the GUI application is loaded. The communication can be done with two or more system connected with one server accessed. The design is done with the two Qt classes Qtcpsocket and Qtcpserver.

### 2.1 Server Section

The server section is used to create a server for a system where clients can give and get data from other system. The GUI design is made with two groups, one to get the mode of communication required and the other is to give a security support for the communication to be handled safe. The mode of selection contain three radiobutton to switch between, which allow to access all port and IP or single port or single IP of single port. The security group requires password to initiate the start of the server.

#### 2.1.1 UI Design – Network Server

With UI design, in the server network group, the allportandip radiobutton is checked by default. The port input and IP input value is restricted to maximum length of 8 and 15. The IP input is used with the input mask value so that the IP can be easily entered in the line.

In the second group, the password is made to display with asterisks as entered, with the special attribute echomode which is set as password. The same is done for verify password part. The password is aligned center both in vertical and horizontal direction.

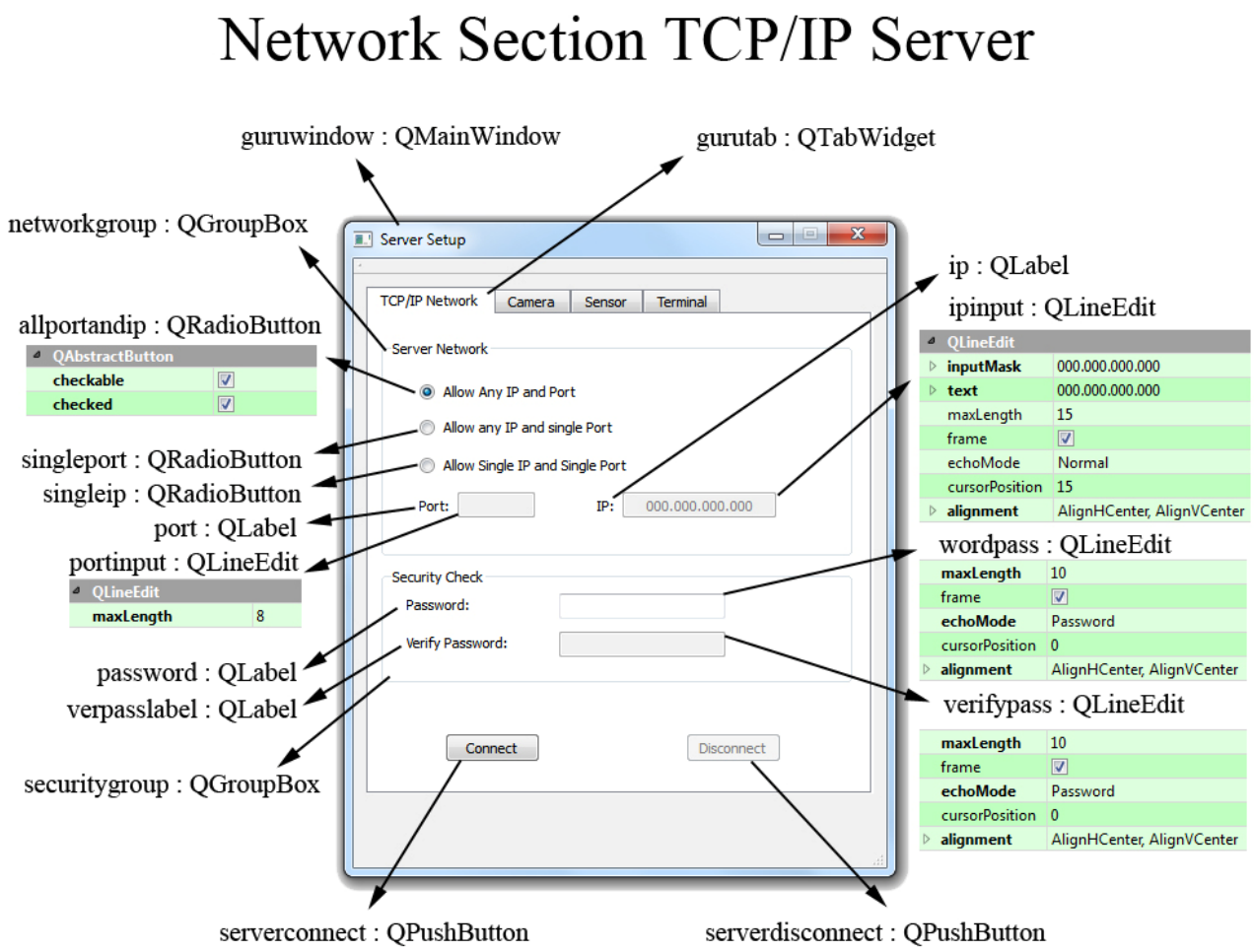


Figure 4: Network section – Server

### 2.1.2 MainWindow Class : guruwindow

**Constructor**

The initiate of the process is triggered with disabling disconnect server pushbutton, input value for IP and port, and verifying password. There is a connection made between the disconnect server button and close server function that when disconnect server button is triggered the server is stopped. Also, there is another connection triggered, when the server is started and there is a new client, which is a safe mode to send sensor data.

**Toggle selection**

The toggle section is designed to enable or disable the IP and port input when there is switch between the radiobuttons.

**Valid Input Check – Port and IP**

When the edit is finished, the function check for the valid input for the port value and IP, if it is not correct the error message pops up and the color of the label is changed to red.

**Valid Input Check – Password**

When the edit is finished, the function check for the password is entered or not. If the QLineEdit is empty then the error message pops up to enter the password and the label changes to red color. If the password is valid, then the value is stored and the verify password is enabled. Here in the verify password, the value entered is compared with the previous password entered and the action is continued with a check for valid or error.

**PushButtons Controls**

The control is given with two push buttons, when the connect server is clicked it checks for all the valid action for the two group and only if all pervious data are entered is correct, the server is started or else the error message is shown for each mismatch. Also, when the server connect push button is clicked there is a message box set to ask the user to start the server, this helps to start the connection safe. The second button is visible only when the first button is clicked and server is started successfully. The second button triggers to disconnect the server and enable the connect button.

There is a safe mode to send data to the client, as only when the server is started successfully and there is a new connection, the data is sent.

### 2.1.3 TcpServer Class : gurutcp

**Start Server**

When the push button is clicked with the valid data, Qtcpserver server is created. The server starts listen to the upcoming clients to be connected. The request for listening is given through the port and IP value during the input section. When a

new client response there is a new socket created and the communication is established.

**Security check**

After the connection, there is a security check, which gives safe communication. The process starts from the client to give “Hello” and the password. If the server password matches then the value is sent back with “ok” else “notok” is sent to the client.

**Data safe mode**

The sensor data is only sent to the client if there is a valid establishment of the communication. Also this is used to stop the program crash, when socket data is sent when there is no client.

## 2.2 Client Section

The client section contains two dialog box, one for the getting the data for the connection – setup client and another for the viewing the data displayed – client page. The setup client contains two group box, one to get IP and Port values and the second group box contains the password check.

### 2.2.1 UI View Diagram – Setup Client

The UI contains input section for the client. Same as the server setup the port input and IP input value is restricted to maximum length of 8 and 15. The IP input is used with the input mask value so that the IP can be easily entered in the line.

In the second group, the password is made to display with asterisks as entered, with the special attribute echomode which is set as password. The same is done for verify password part. The password is aligned center both in vertical and horizontal direction. The pushbutton is used to trigger the start of the client.

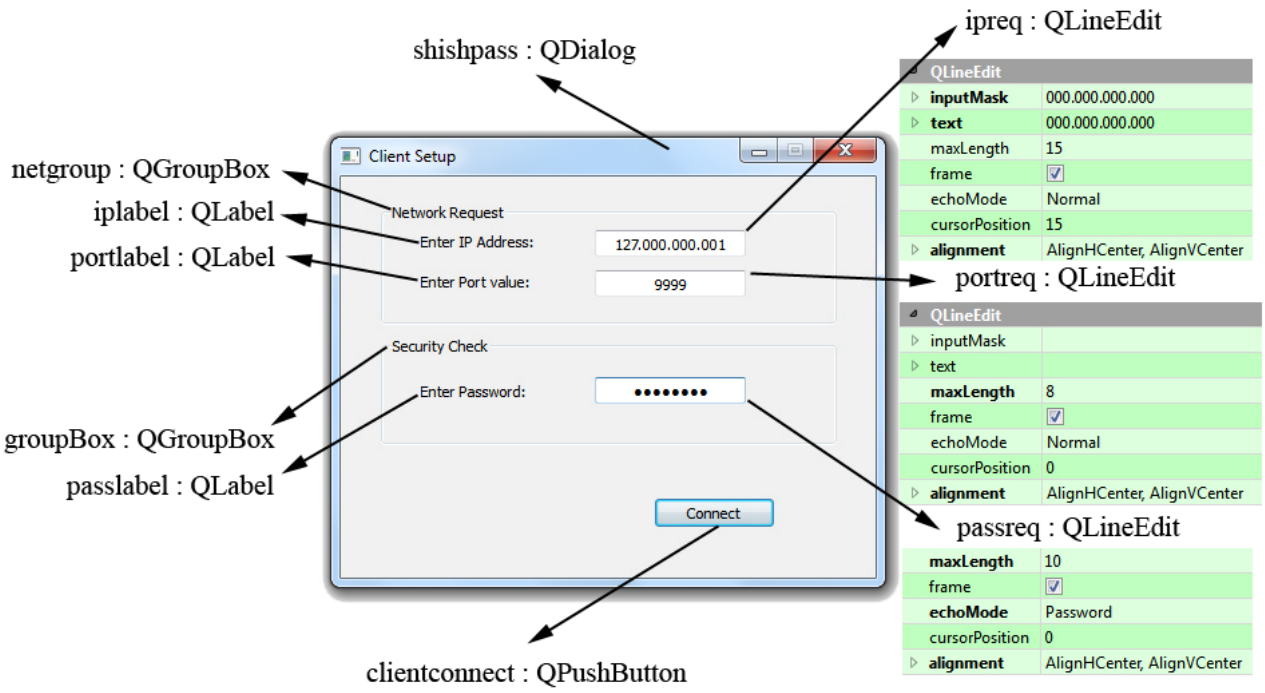


Figure 5: Network Section – Client

### 2.2.2 UI View Diagram – Network Client Page

The UI is used to display the 3D display and the console of the data flow. Labels are used to indicate the Port and IP value of the connection. The push button is used to disconnect the client.

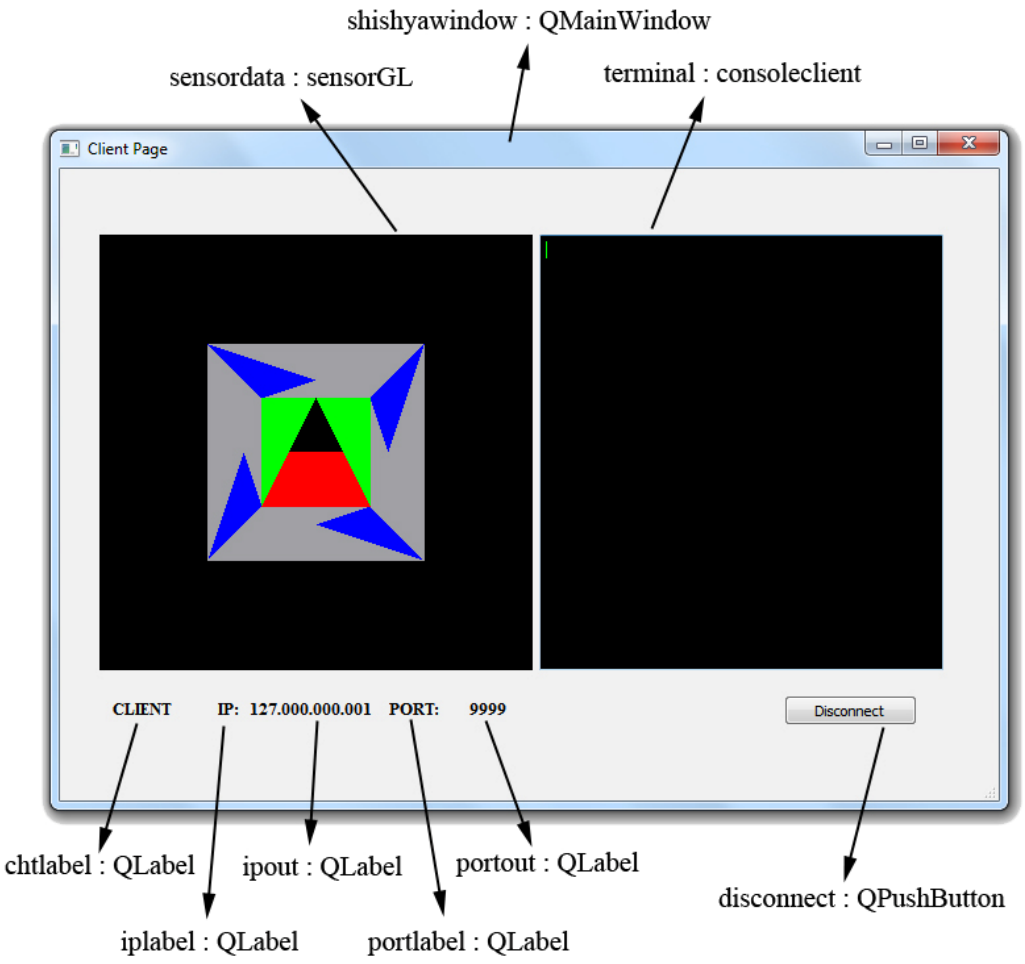


Figure 6: Client Page

### 2.2.3 Tcpclient Class : shishpass

**Constructor**

The constructor contains the general specification for the window setting the name for window and making the size fixed. The connections are made between received sensor value and display page. There is an another connection to close the connection.

**Edit check**

The values for the IP, Port and password are check in the section. The error message and label of the corresponding will change to red if there is some mistake in the input.

**Push button action**

The push button trigger to do a multi-check and then if all the input are correct, the client socket is created. The welcome data is sent with the password. If the input is valid with correct server password, then the connection is established and the server starts listening. The client setup window is closed and the client page is opened for the sensor display.

**Read sensor**

The received sensor data is collected and then they are filtered with parameters. The filter process is done with the axis of the data and sent to the respective function to display.

**Disconnect**

When the disconnect push button is clicked, the socket is closed with the window and opens the setup window for entering now data.

### 2.2.4 MainWindow Class : shishyawindow

The class acts intermediate between the client page and the ui of the class. The data and close action for client are communicated through this class. In the display section the 3D view is shown in the left part which is promoted by sensorGL class and the console is shown in the right part which is promoted by the consoleclient class.

# CHAPTER 3

## 3 CAMERA SECTION

In this section, the camera available in the system or server computer is identified and listed to be started. The process is done with the help of QCamera and QCameraViewfinder class.

### 3.1 UI Design – Camera Setup

The UI design contains promoted widget of QCameraViewfinder and group box to access the control for the camera selection.

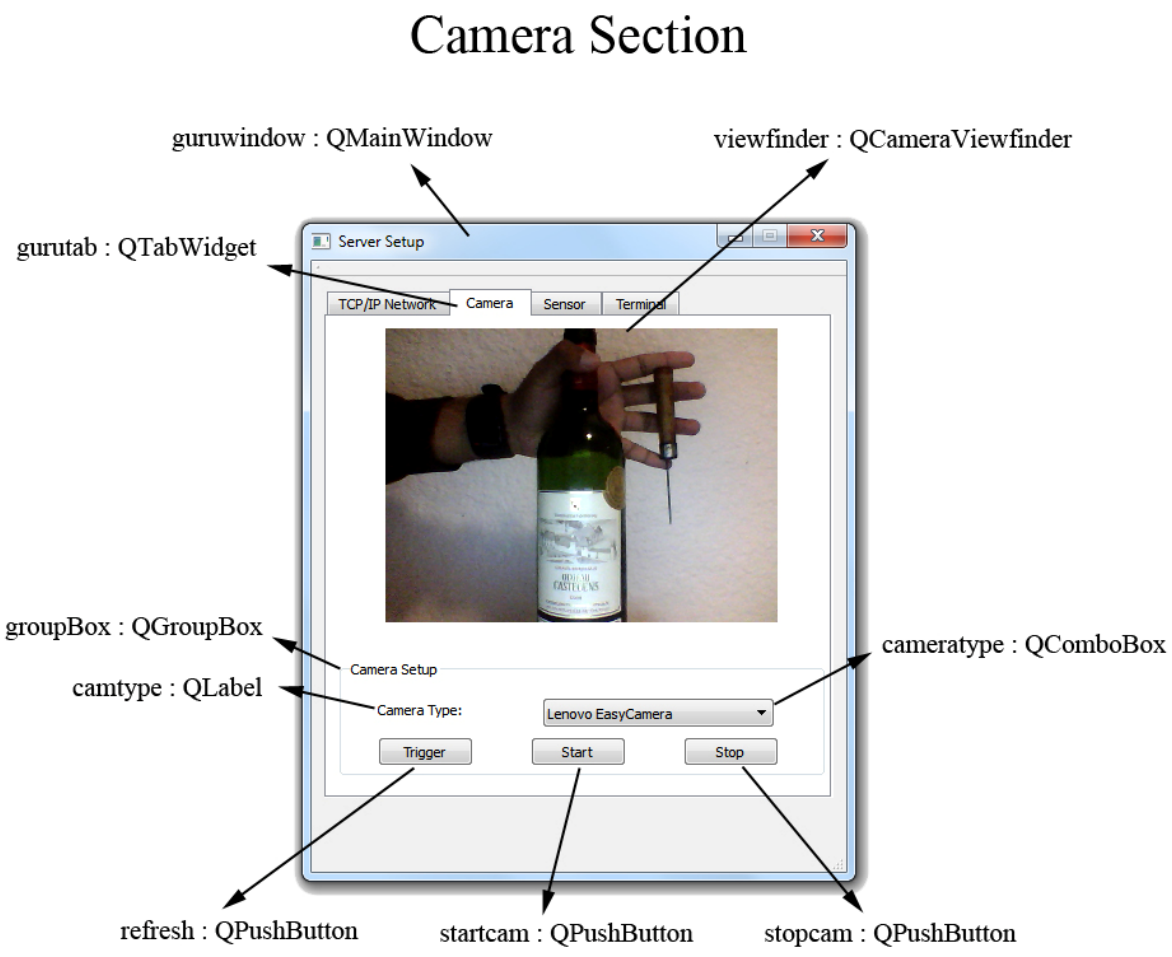


Figure 7: Camera Section

The program for camera setup is done in the guruwindow class. In the constructor, the initial setup is disabled for all the control to start the safe start.

### 3.1.1 Pushbutton

The refresh pushbutton clears all previously added combo box item (if already added), and checks for the available camera device and add it to the combo box item every time when the button is clicked. The device is set with the first identified device and the setting is made for that device. The camera is controlled with the start and stop push button. The process is switched between then when they are clicked.

### 3.1.2 Camera setup

When the new device is selected from the combo box the index is checked and that particular device name is passed to the setup function. The selected device is called and then displays the image in the QCameraViewfinder. When there is a error detected it is shown as warning message box.

## CHAPTER 4

## 4 SENSOR SECTION

The sensor section is 3D display of the sensor value using OpenGL. The section contains two parts one is the promoted class of sensorgl class which contains the entire openGL program and second is the test the 3D display with manual input of the values.

### 4.1 OpenGL Section

The program to design the 3D model is done in sensorgl class and in guruwindow class the pushbuttons are used to start and stop the sensor.

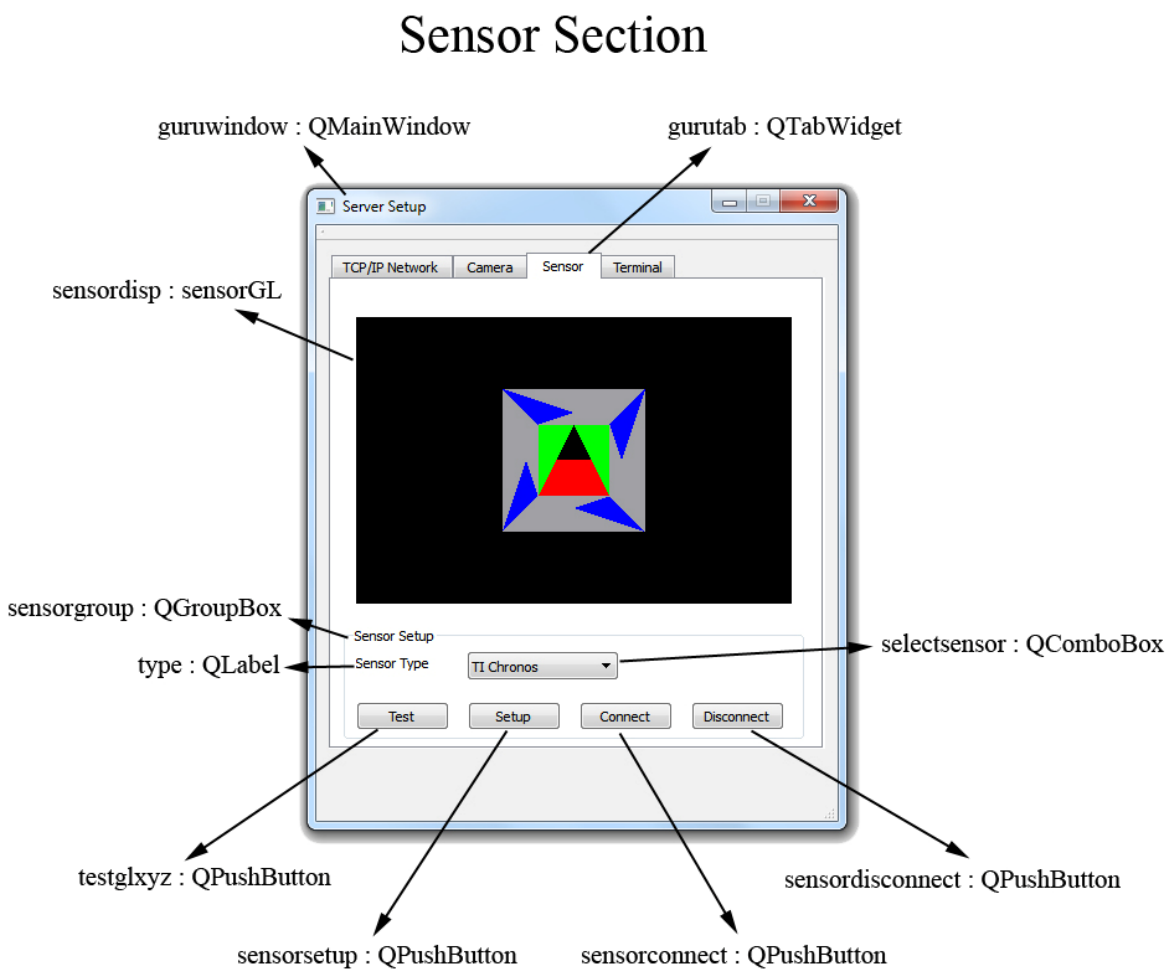


Figure 8: Sensor Section

### 4.1.1 UI View Diagram – Sensor section

The UI Design contains the display of the 3D design in the top. Sensor type is used to select various sensors that can be used with the software. At the bottom there are pushbuttons to open the test dialogbox and buttons to set, start and stop ti sensor.

### 4.1.2 MainWindow Class : guruwindow

**Constructor**

The constructor connects the data communication between real and test sensor values to the 3D display. All the pushbuttons are disabled first and only when the terminal is successfully started the buttons are enabled.

**Sensor selection**

The sensor is selected here, if the ti sensor is used then the pushbuttons are enabled and the control is used only with the buttons and terminal is disabled from input value. When other sensor is selected then the parameter data is communicated only with the terminal for both the directions.

**Pushbuttons**

The pushbuttons are used to control the sensor access and to open the testgl dialogbox. The pushbutton is designed for the better access for communication with the ti sensor watch.

### 4.1.3 Sensorgl class

When the object is created,the class in promoted in the guruwindow class. This calls initializeGL, paintGL and resizeGL automatically to display the model. The draw function is called to initial all the vector points and the angle of the light displayed.

### 4.2 Test Section

In this section, the sensor data is manually passed to the display and to the network client. The design of the dialogbox contains three sliders with the value 0 to 360 for the angles. The testgl class emits the data value of as a signal.

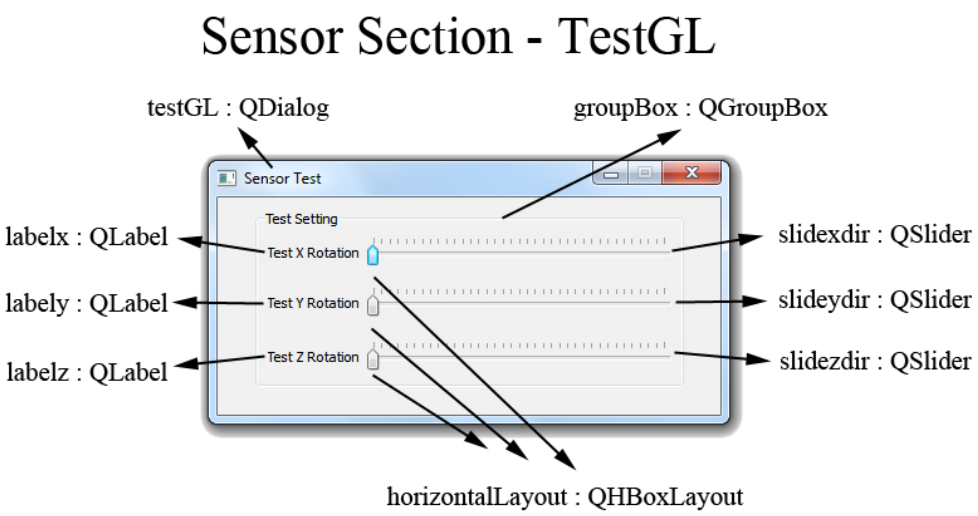


Figure 9: TestGL Dialogbox

## CHAPTER 5

## 5 TERMINAL SECTION

In this section, the setting for the sensor is managed and the data that are communicated can be viewed here.

### 5.1 Console Section

The console section is a promoted class of consoledisp and placed in guruwindow class. The serial communication is accessed here, with the input from the sensor settings.

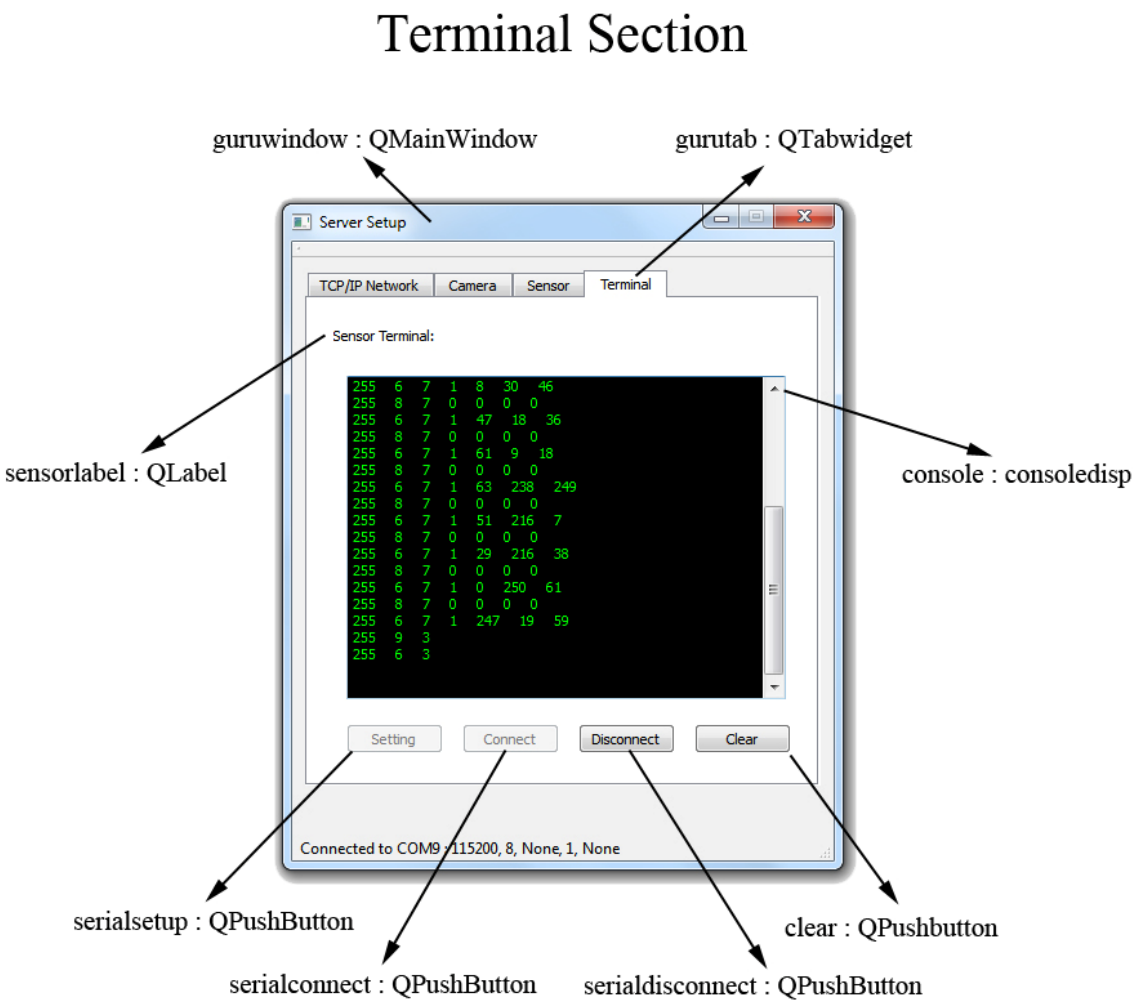


Figure 10: Terminal Section

### 5.1.1 UI View Diagram – Terminal

The UI design contains a simple sensorlabel and followed by the display of the console terminal. The pushbuttons are used to start and stop the serial communication port values.

### 5.1.2 MainWindow Class : guruwindow

**Constructor**

The sensor communication data are taken from the setting dialogbox and are sent to the qtserialport for creating the connection. The pushbuttons and the console is disabled first until a new connection is established.

**Pushbutton and Control**

The pushbuttons are used to create, start and stop the serial communication. The control of error handling is done automatically, as when there is an error in starting the serial port, the error is handled and error message is displayed. Opencontrol function is called to enable and disable the sensor data in the sensor section.

### 5.1.3 Consoledisp class

The class is used to give and get the sensor value and also to be displayed. The palette is displayed with black color background and green color text. The print session is managed here.

## 5.2 Sensor setting section

In the section, the serial communication data is asked from the user to connect. The available port are scanned and displayed in the serial port info list box so that the required com port can be selected.

### 5.2.1 UI View Diagram – Sensor setting

The UI design contains three group box, one to select the com port, second to select the value for the communication and third to set to echo mode. The apply button sends the selected value to start the serial communication with the selected port.

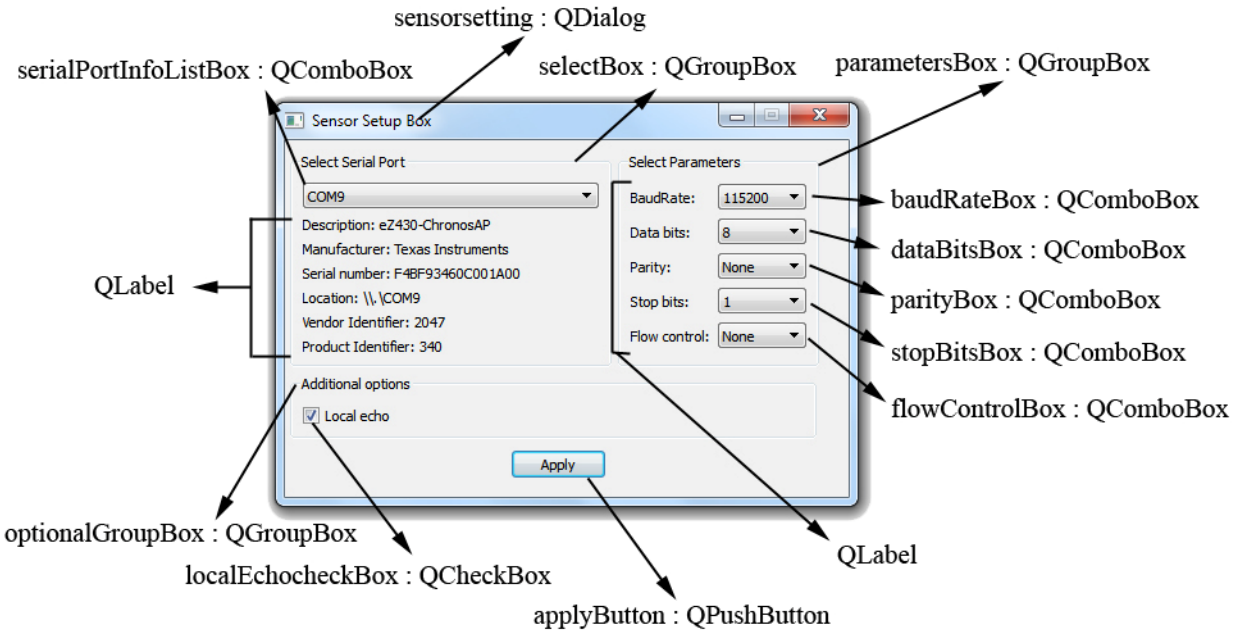


Figure 11: Sensor Setting

### 5.2.2 Sensorsetting class

The sensorsetting class fills all the value for the selection parameter. The selected com port descriptions are displayed in the labels later. The pushbutton closes the dialogbox and sends the parameter selected to initiate the communication.

## CHAPTER 6

## 6 ACTION SECTION

In this section, the menus are added and perform the specific action when clicked. The toolbar are displayed with the icons for easy access of the software application.

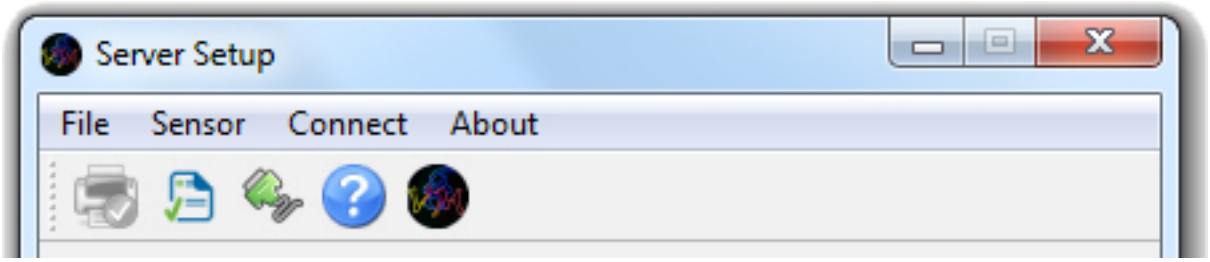


Figure 12: Action Section

## 7 TEST SETUP

To test the application, virtualbox software is used to communicate the software between different platforms. Linux is installed and the network is configured to access the data through tcp/ip.

**Some problems faced:**

| Problem: | Solution: |
|---|---|
| Screen resolution | sudo apt-get install virtualbox-guest-dkms |
| Could not initialize GLX | sudo apt-get install libgl1-mesa-dri |

## 8 BUILDING STATIC APPLICATION

To make the static application, QT should be built static. The step to build are, install Python, Perl and Ruby software. Inside the source folder, open qtbase and chose the compiler. The tested environment is with ‘win32-msvc2010’ which is inside mkspecs folder.

Edit,

```
QMAKE_CFLAGS_RELEASE    = -O2 -MD
QMAKE_CFLAGS_RELEASE_WITH_DEBUGINFO += -O2 -MD -Zi
QMAKE_CFLAGS_DEBUG      = -Zi –MDd
```

To,

```
QMAKE_CFLAGS_RELEASE    = -O2 -MT
QMAKE_CFLAGS_RELEASE_WITH_DEBUGINFO += -O2 -MT -Zi
QMAKE_CFLAGS_DEBUG      = -Zi -MTd
```

Open Microsoft Visual Studio command prompt, and go to the qtbase.
The command to make qmake,

**→configure -debug-and-release -opensource -platform win32-msvc2010 -opengl desktop -static -nomake examples -nomake tests**

To compile,

**→nmake sub-src**

the project is make static and compiled with the static qmake file.

## 9 FUTURE DEVELOPMENTS

The application can be taken to next level by increasing the accessibility of data flow. The project can be developed to,

- Communicate with multiple sensors.
- Additional features for image processing.
- Start the application when the OS is started.
- Saving the data in Database.

## CONCLUSION

The software is a simple protocol for communicating between different robot modules. The software is in starting stage to make the communication better. The scope for extension of the application is better, to make use of the software in the automated system. The software is useful in simple data exchange and data tracking.

**REFERENCE**

1) QT Example is used as a great source of reference to design the application.
http://doc.qt.io/qt-5/qtserialport-terminal-example.html

2) For the ti sensor,
http://processors.wiki.ti.com/index.php/EZ430-Chronos
http://e2e.ti.com/support/microcontrollers/msp430/f/166/t/32714

3) For Building the application static,
http://stackoverflow.com/questions/14932315/how-to-compile-qt-5-under-windows-or-linux-32-or-64-bit-static-or-dynamic-on-v
https://wiki.qt.io/Build_Standalone_Qt_Application_for_Windows

4) Online QT Tutorial
http://www.bogotobogo.com/Qt/