Abstract - Remote monitoring of home or office is much needed application in terms of security especially on mobile platform. Monitoring baby for parents or watching over empty house when user is away for vacation are few examples where such applications can be used. Detailed weather reporting which includes temperature, humidity, rain, and wind direction, speed along with storage of historic weather data can helpful for user for future weather prediction. This project developed an application on IOS platform which provides user capability of monitoring and controlling Wi-Fi enabled camera. Also, the project interfaced weather reporting sensors to Arduinos to collect and view weather information.

1. Introduction

Arduino [citation/reference] is an open source single hardware board that consists of Atmel 8 bit AVR microcontroller. Arduino Mega 2560 as well as Mega ADK board has 256 KB of flash memory. Both have 54 digital input/output pins (of which 14 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. Power can be supplied using USB connection or external supply. Manufactures provide an integrated development environment (IDE) which can be used to write software for Arduino and upload it on the board using USB connection. This IDE can be used on Windows, Mac OS X and Linux. Arduino official site [citation/reference] provides detailed information on programming on this IDE. Arduino can also be connected to external Ethernet or Wi-Fi shield to provide its local IP address and hosting small web server. Personalized weather sensors are connected to Arduino, which calibrates value and posts it on the web server.

An open source IP camera which when powered on acquires local IP address and can be viewed remotely using this IP address. This IP camera has additional functionality where in user can control motion of the camera in various directions. This functionality can be explored and provided to user for remotely viewing camera in various directions. Most common format for video on these web cameras is MJPEG, which is supported on IOS device.
2. Related Work

There are some applications developed [reference/citations] that communicate with Arduino, receiving and sending data from Arduino using IOS device. Arduino Manager [reference/citation] is application released in App Store, which controls various pins of Arduino through Ethernet shield. Other applications, such as IP Cam Lite [reference] provides remote viewing for web camera and also allows for controlling web camera position. As well as there are some applications like Meteo Box, which provide real time temperature, humidity and pressure information. This application however does not provide historic data and is not been updated for IOS 6.

The main motivation for this project is to learn about the technologies integrated in this project, including Arduino, iOS applications, and the various weather sensors. The project problem is to provide mobile iOS application with the following functionalities, remote viewing of instantaneous weather information, summary plotting of weather information, viewing web camera video, instantaneous control of web camera motion (up, down, right left), as well as web camera snapshot capture, snapshot storage and review. This mobile application provides remote camera viewing as well control, along with real-time weather information and graphical charts of past weather data.

3. Approach

Temperature and Humidity Sensor

There are various composite temperature and humidity sensor available in the market. This project makes use of DHT22 [reference] a low cost temperature and humidity sensor. This sensor has four pins, of which three needs to be connected to Arduino. First pin is connected to 5V pin of Arduino. Second one is data pin, which can be connected to any digital pins of Arduino, and it needs to be in series connection with 4.7K resistances and 5V power supply. Third pin is not connected and last pin is connected to the ground. The temperature and humidity value is read by Arduino continuously and posted on the web server. Pictorial representation of the connection is shown below.

Figure 1: Pin Connections for DHT22 sensor with Arduino
Weather Meter

Rain, wind speed and direction calibration is done using weather meter provided by Sparkfun Electronics[ reference]. None of the sensors in this kit contain active electronics, instead they use sealed magnetic reed switches and magnets so one need to source a voltage to take any measurements. The positive side of this is that the sensors are easy to interpret. All of the included sensors are supplied with RJ11 terminated cables and need additional RJ11 pin out jack, which can be used to connect it to the breadboard. The pictorial representation of the weather meter is provided in Figure 2.

Figure 2: Weather meter from Sparkfun Electronics (SEN-8942)

The anemometer (wind speed meter) encodes the wind speed by simply closing a switch which each rotation. A wind speed of 1.492 MPH produces a switch closure once per second. This is connected to Arduino ground on one side, and pin 2 is interrupt pin. Pin 2 is pulled up, and the reed switch on the anemometer will send signal to ground once per revolution, which will trigger the interrupt. The number of revolutions are measured in 5 seconds and divided by 5.

One Hz (rev/sec) = 1.492 mph.

The wind vane reports wind direction as a voltage, which is produced by the combination of resistors inside the sensor. When a voltage is supplied, the voltage returned can be translated to any of 16 possible positions. We use a classic voltage divider to measure the resistance in the weather vane, which varies by direction. Using a 10K resistor, our ADC reading will be:

\[ 1023 \times \frac{R}{10000+R} \]

Where R is the unknown resistance from the vane. Figure 1023 is scaled down to a 255 range, to match the datasheet.
Depending on orientation of the system, direction needs to be adjusted.

<table>
<thead>
<tr>
<th>Sector</th>
<th>Reading</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>18</td>
<td>W</td>
</tr>
<tr>
<td>1</td>
<td>33</td>
<td>NW</td>
</tr>
</tbody>
</table>

The rain gauge is a self-emptying bucket-type rain gauge that activates a momentary button closure for each 0.011" of rain that is collected. It is too connected in similar fashion as anemometer. Pin one is connected to ground of Arduino and pin 3 is interrupt pin. Each interrupt represents .011 inches of rain (0.2794 mm). Both anemometer and rain gauge do not need additional power supply.

The wind speed, direction and rain is calibrated by Arduino and posted on the webserver.

**Remote Web camera Viewing and Control**

This project uses FOSCAM FI8918W model of web camera for remote viewing. Any open source camera, which makes provides IP address for viewing it through browser will serve purpose of the application.

Application uses **UIWebView** class to view the broadcast of the web camera. These cameras require authentication to view the feed. The feed can be directly viewed using URL http://IPADDRESS/videostream.cgi?user=[user]&pwd=[password]. Application makes use of loadHTMLString method of UIWebView class for providing live stream where string embeds live feed tag in the string. The html string is of the following format:

```
"<img name="Cam" src="%@" width="100%" height="100%" alt="Live Feed" style="background-color: #000000;" />"
```

Where src is http://IPADDRESS/videostream.cgi?user=[user]&pwd=[password].

This camera can be moved up, down, left, right, center, rotate in horizontal and vertical axes. This can be done externally by sending HTTP GET request to the camera with appropriate control parameter for respective motion. The application uses sendSynchronousRequest method of NSURLRequest class to send the GET request. It should receive OK response from the camera if the necessary action is performed. The GET request URL is of the form - “http://IPADDRESS/decoder_control.cgi?command=[control]&user=[user]&pwd=[password]"

Following is table for control number for movement –

<table>
<thead>
<tr>
<th>Number</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>UP</td>
</tr>
<tr>
<td>2</td>
<td>DOWN</td>
</tr>
<tr>
<td>4</td>
<td>LEFT</td>
</tr>
<tr>
<td>6</td>
<td>RIGHT</td>
</tr>
<tr>
<td>25</td>
<td>CENTER</td>
</tr>
</tbody>
</table>

The application also provides way to change image orientation from camera. Video can be flipped, mirrored and flipped as well as mirrored together. Application sends HTTP Get request to the camera to perform necessary action. If request is serviced by
camera, application should receive OK response. The GET request URL is of the form –

http://IPADDRESS/camera_control.cgi?param=5&value=[value]@&user=[user]&pwd=[password]

The table below gives value corresponding to camera orientation –

<table>
<thead>
<tr>
<th>Orientation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Default</td>
</tr>
<tr>
<td>1</td>
<td>Flip</td>
</tr>
<tr>
<td>2</td>
<td>Mirror</td>
</tr>
<tr>
<td>3</td>
<td>Flip + Mirror</td>
</tr>
</tbody>
</table>

It is also possible to take snapshot of current video stream. The functionality is implemented using UIGraphics support for IOS. The current rendered image data is stored in UIImage class. This image is later converted in PNG format, which is popularly supported by IOS device. This image is stored in documents directory with timestamp as name of the image. Application allows user to change camera settings or take snapshot only in the video is streaming on the screen. Below is application screenshot of the all the functionality described –

**Figure 3: Screen shot of application page to view web camera broadcast and control camera settings**

**Photo Gallery and Email Option**

Application allows user to browse through the gallery of the images taken from the live stream. Application supports left and right swipe gestures, which are implemented using UISwipeGestureRecognizer class. User left and right swipe gestures are captured and pictures are moved left and right. All these images are maintained in Documents directory. Application also provides feature for user can to send image to someone through email. Requirement of this feature is that user has mail account setup on the device. Photo gallery and email option are an add-on features to maintain snapshot of video stream.
Interaction of IOS Application with Weather Station

View Controller class is first screen user interacts with on the application. It takes as input IP address of web server that is posting values of temperature, humidity and weather meter. It creates in background timer task, which goes and poll the webserver for temperature, humidity, wind speed, wind direction and rain value every hour the application is running. The polling request is HTTP Get request to server. The values obtained will be updated on the application.

SQLite database

Prior SQLite database needs to be created and copied into resource folder. Since there is going to be modification to this database, it is essential to copy this database from application bundle into device document directory. This is just one time action that needs to be performed and is generally done during applicationdidFinishLaunching method of the AppDelegate. There are three different queries that application performs -

A. Weather at different time intervals for a particular day: This query for average temperature and humidity for every hour.

```
e.g. "select * from records where strftime('%d', date) = '24' and strftime('%m', date) = '04' and strftime('%Y', date) = '2013' where time = '11' ORDER BY time"
```

B. Average weather for all days for a particular month of year: This query needs to calculate average for each date of that month.

```
e.g. "select date, avg(time), avg(temperature), avg(humidity) from records where strftime('%Y', date) = '24' and strftime('%m', date) = '04' and strftime('%d', date) = '2013'"
```

C. Average weather for all the months of a particular year: This query calculates average readings for every month.

```
e.g. "select date, time, avg(temperature), avg(humidity) from records where strftime('%m', date) = '04' and strftime('%Y', date) = '2013'"
```

The each value returned is stored in object WeatherRecord and the array of these objects is returned back. If the query returns nil, no graph will be plotted. If returns objects back, graph will be plotted based on these values.

Graph Plotting

Core plot library is used for plotting graphs based on the database query. Application needs to include necessary core plot files
and library for IOS platform in the framework. Both temperatures as well as humidity are plotted on the same graph. Application needs to setup axes, labels, legends, data points for every new query of the user.

Settings

The application maintains settings page separate from the other screens to maintain information about IP address of the web server, weather station, temperature and humidity sensor. User can go and edit the settings anytime there is change in IP address and application will accordingly be able to pick correct IP address to poll for weather information or stream the broadcast.

![Figure 5: Screenshot of settings screen of application](image)

4. Problems

Initial approach for camera movement was using servos, which will be connected to additional Arduino. Arduino was supposed
to receive commands from the IOS device to change servo positions, which in turn would change orientation of the camera. Application was successful in establish communication with Arduino and exchange messages. However when servos where connected to Arduino along with Wi-Fi shield, Arduino was able to create socket but was not able to bind to incoming socket connection from the device. The root cause of this issue was not been able to figure out. With additional in built functionality of the FOSCAM web camera to pan tilt, it was decided to explore this camera functionality on IOS device.

5. Future Work

The project can be expanded into big scale where there can be more than one web cameras connected and user is able to monitor more than one location at one time and control the camera at these various spots. Similarly weather reporting can be expanded to more than one location and user is able to view current time weather information of various locations as well as see historic data region wise and compare them.

References

http://www.arduino.cc/
https://sites.google.com/site/lurvill/arduino
manager_1-6
https://www.sparkfun.com/products/8942
https://www.sparkfun.com/products/10167
http://www.sqlite.org/
https://code.google.com/p/core-plot/