Zack Whittaker and Xudong Qin
ELE 408
May 2, 2019

Web-Enabled Remote Sensing for Nuclear Applications

Motivation

After an event like the 2011 Fukushima nuclear disaster, the area around a power plant can reach levels of radiation much too high for humans to navigate. Therefore, post-mortem inspection and cleanup of the area is most often handled by specialized robots. These robots are built to navigate the uneven remains of the powerplant and send back important data such as video, temperature, and radiation levels. Since human operators must stay out of areas of the highest radiation, the ability to control the robots over the web is key to maintaining operation.

Project Objectives:

- Deploy a remotely-operated robot into a room to simulate a disaster site
- Control the robot via the DE-1 touch screen
- Collect real-time data on the robot and send to the DE-1
    - Video streaming
    - Temperature, humidity, and pressure

Prerequisite:

- **Programming language:** Python, javascript, HTML, sqlite command line.
- **Hardware:** Raspberry Pi 3, SenseHat, Osoyoo robot car, DE-1 board
- **All the codes are available on GitHub:**
  https://github.com/XD-QIN/ELE408FinalProject

Design:

This project was based on the Osoyoo Raspberry Pi Robot Kit and a Raspberry Pi 3 B+ kit from Canakit. The robot kit contains the chassis, motor controller board, motors, camera, and five digital light sensors. Also included is an SD card loaded with the Raspbian Operating system and pre-installed packages such as WebIOPi and mjpg-streamer.

To add sensing capability, we purchased the Raspberry Pi SenseHat which mounts on top of the GPIO pins of the Raspberry Pi and communicates with the Pi via I2C. This was necessary since the Raspberry Pi does not have a built in A/D converter like the FPGA. This sensor has open-source API in both Python and C++ which made interfacing with the sensors very simple.

The last piece of hardware was the router which was needed to ensure that the servers and clients would not be affected by the University network settings. This also allowed us to give the robot and FPGA a static IP address that could be known without connecting a display and checking the output of ifconfig each time.

The final product was a system that allows for remote control and video streaming through a web browser, live data monitoring via TCP socket, and long-term data collection accessible through networked SQLite requests. Figure 1 shows the system design for each of these features:

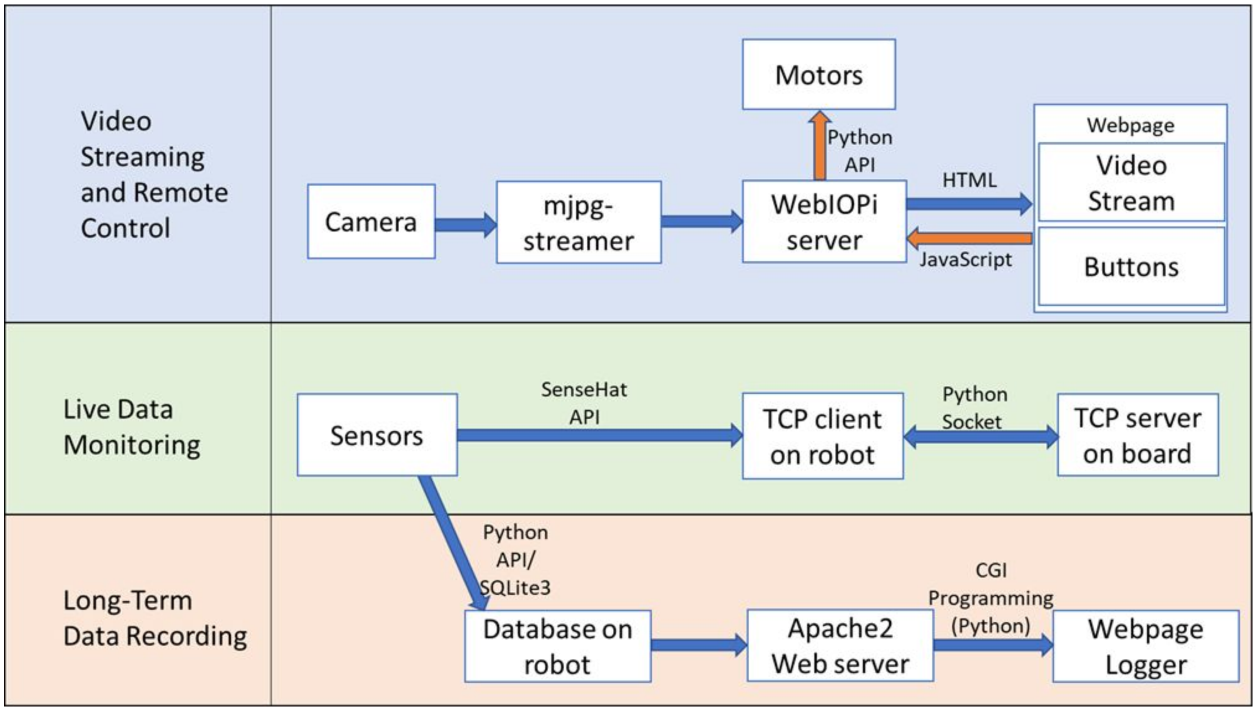


Figure 1. System Components and Interfaces

Tutorial:

This section will teach you how to run the demo assuming you have purchased all of the hardware from the design section and followed the tutorials provided by Osoyoo to set up the robot car.

Video streaming and remote control:

If using the pre-formatted SD card provided with the Osoyoo Robot Car Kit, there are a couple of ways to start the video streaming and remote control servers.

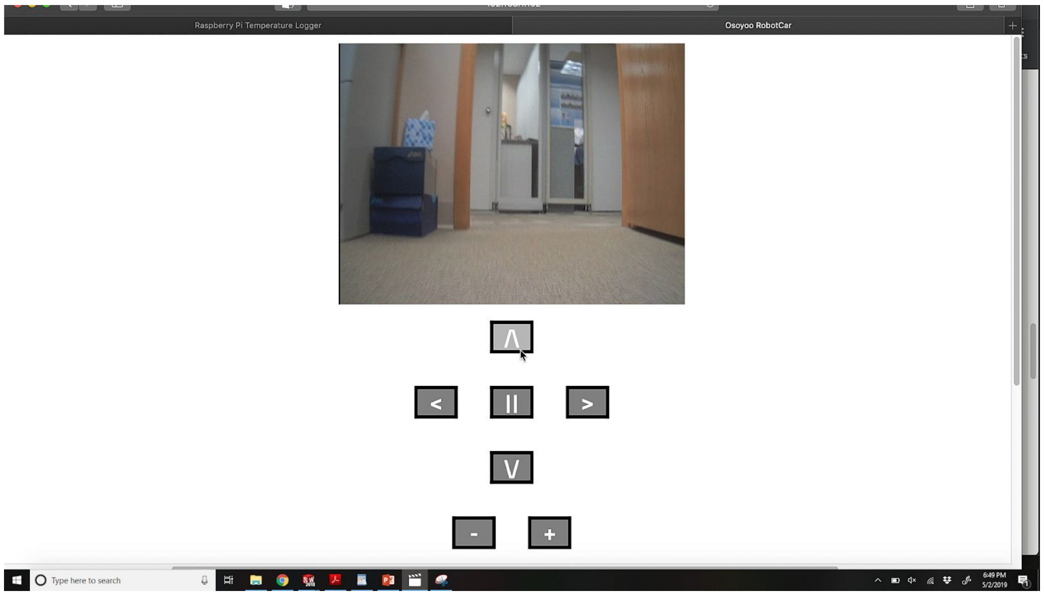
1. Run the startdemo bash script
    a. Click on the startdemo.bsh script on the robot desktop and select “run in terminal” or open a terminal on the Desktop and type sh ./startdemo.bsh
2. Open two terminals on the Raspberry Pi using keyboard and display, ssh from another computer, or using VNC connect to show the deskop
    a. In first terminal, cd to /home/pi/osoyoo-robot/cam-robot/mjpg-streamer
    b. Type sh ./start.sh
    c. In the second terminal type: webiopi -d -c /etc/webiopi/config

This will start the mjpg streamer which sends the images from the camera to the webpage at http://your-RaspberryPi-ip-address:8899 and the webiopi server that displays the video feed and the buttons to control the robot on another webpage at http://your-RaspberryPi-ip-address:8000

To view the webpage, navigate to http://your-RaspberryPi-ip-address:8000 on another computer and type in the following at the prompt:

Username: webiopi
Password: raspberry

If all was successful, the webpage should look like the following and the stream should show the feed in real-time. Additionally, if the robot is running on battery power, clicking the buttons on this webpage should let you control the robot.



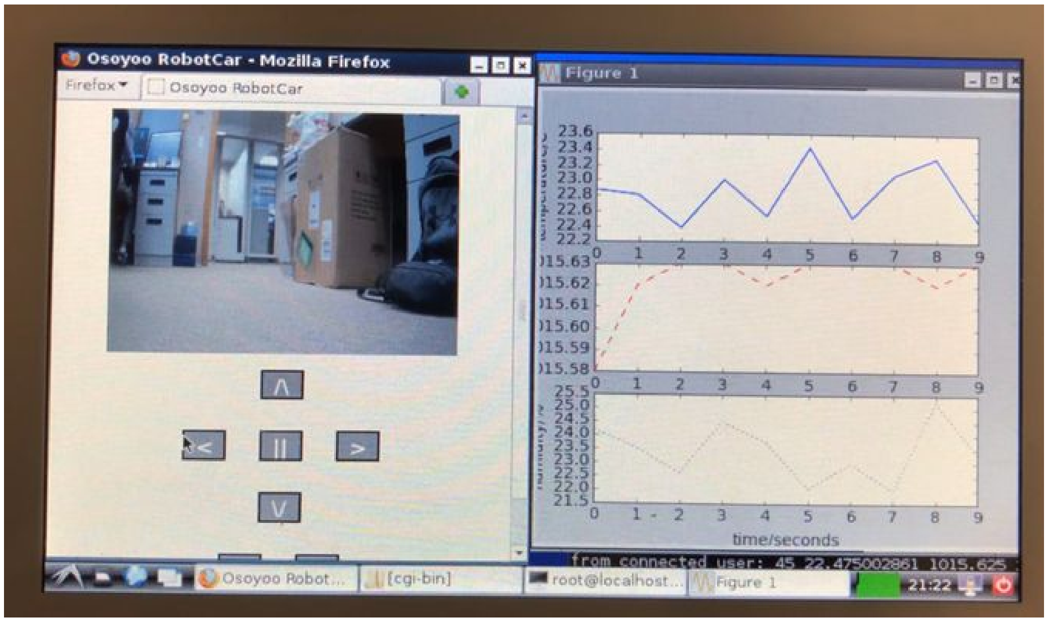
Real-time data Monitoring:

Before we start, we first need to change the ip address of line 22 in python script Final_Project_FPGA_TCP_server.py to the ip address of your own DE-1 board and similarly change the ip address of line 11 in file dataoversocket.py to DE-1 board. After finishing the previous ip address setting step, all you need to do is the following two steps:

1. On DE-1 board: $ python Final_Project_FPGA_TCP_server.py
2. On Raspberry Pi: $ python3 dataoversocket.py

Then you will see a window prompt on DE-1 board touch screen and begin to plot the real-time data in the last 10 seconds.

If you successfully implement the previous, you will see the following picture on the DE-1 board.



```
from connected user: 45 22.475002861 1015.625
```

Long-Term Monitoring:

In this part, we will build a database based on sqlite3 to store the data collected from SenseHat. Then we build a website based on Apache2 to show all the data.

1. Update raspberry pi: $ sudo apt-get update
2. Install Apache2 web server and enable the cgi-bin scripts
   2.1. Install Apache2: $ sudo apt-get install apache2
   2.2. Enable cgi-bin scripts: $ a2enmod cgid
   2.3. Restart the web server: $ service apache2 restart
3. Install sqilte3 database and build a database for a temperature logger
   3.1. Install sqlite3: $ sudo apt-get install sqlite3
   3.2. Create a database: $ sqlite3 templog.db

In sqlite3 command line input the following command:

```
BEGIN;
CREATE TABLE temps (timestamp DATETIME, temp, NUMERIC);
COMMIT;
.quit
```

3.3. Copy database to the /var/www/ folder and change the owner of this database :

```
$ sudo cp templog.db /var/www/
$ sudo chown www-data:www-data/var/www/templog.db
```

3.4. Copy monitor.py and webgui.py to /usr/lib/cgi-bin/:

```
$ sudo cp monitor.py /usr/lib/cgi-bin/
$ sudo cp  webgui.py /usr/lib/cgi-bin/
```

3.5. Make webgui.py and monitor.py executable:

```
$ chmod 777 /usr/lib/cgi-bin/monitor.py
$ chmod 777 /usr/lib/cgi-bin/webgui.py
```

3.6. Create a time-based job in the linux system:

```
$ crontab -e
```

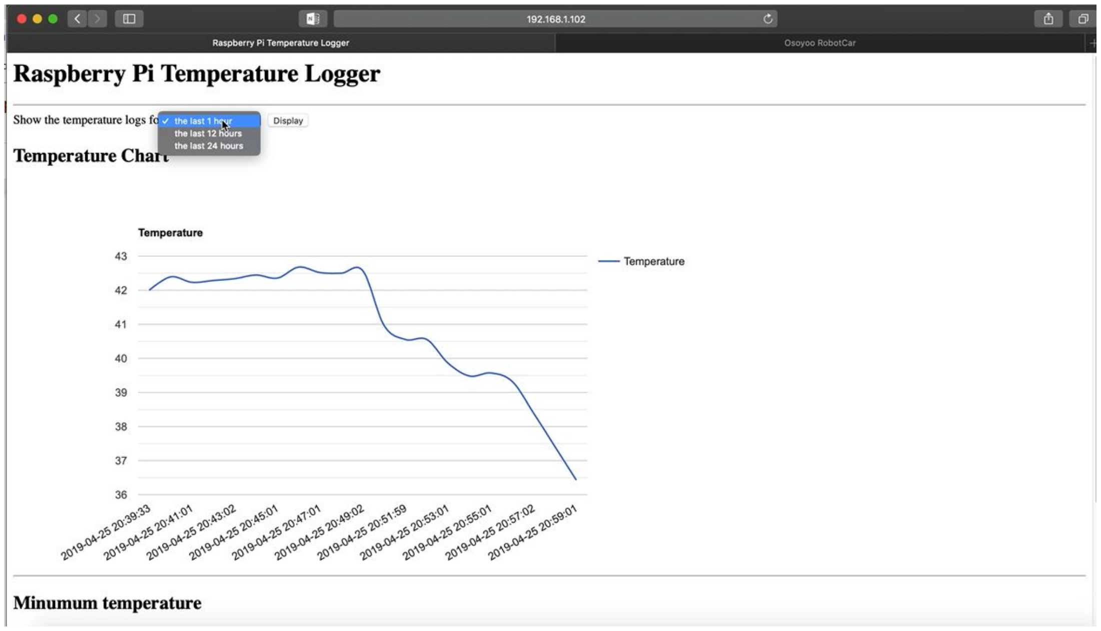
Insert the following line:

```
*/1 * * * * /usr/lib/cgi-bin/monitor.py
```

Then save and quit. This step will create a job execute code monitor.py every one minute and store the data in the database

3.7. Use any web browser and visit the following link :
http://your-RaspberryPi-ip-address/cgi-bin/webgui.py

3.8. If you succeed in doing the previous steps, you will see the following webpage:



Raspberry Pi Temperature Logger

Show the temperature logs fo[r] the last 1 hour

Temperature Chart

Minumum temperature