Project Requirements Document

Team: J VANT

Project: ResQ Sensor

Authors:

Vivian Escalante vivianescalante@umail.ucsb.edu

Alexis Jimenez alexisjimenez@umail.ucsb.edu

Noura Kiroloss nkiroloss@umail.ucsb.edu

Fengyu (Ted) Wang fengyuwang@umail.ucsb.edu

Yuxiang (Jim) Zhu zhu@umail.ucsb.edu

Introduction:

Our goal is to develop an app, ResQ Sensor, using the FLIR Lepton, a thermal imager, that will keep your home, your business, and your family safe and secure without your constant supervision.

The Lepton will essentially be used to detect undesirable temperatures as inputted by the user in order to detect unwanted actions. A user will be given the option to input a temperature range for the Lepton to detect. If the Lepton detects this temperature, an action will then take place depending on the use of our product. In our example, we will set our product to detect fire temperatures, such as a candle that remained lit while no one was home (approximately 1,400 degrees celsius). In this case, a signal will be sent to the Raspberry Pi causing an extinguisher to set the flame off. For our purpose, the extinguisher will be a water pump that will shoot water at the flame to put it out.

In order to implement these types of actions, we will have to create a user interface allowing the insertion of temperatures, program the lepton to detect these undesirable actions, along with the extinguisher to work as an independent system aimed to only affect the target area.

Our finished product can then serve other purposes such as detection of home intruders with the temperature set to the average human body temperature or even in the military for the same purpose.

Glossary of Terms:

- **Raspberry Pi**: a single board computer system
- **Lepton Thermal Imager**: an ultra-compact thermal imaging camera
- **Thermal Imaging**: technique using heat given off by an object to produce an image of it
- **GPIO (General-Purpose Input/Output)**: a pin on an integrated circuit whose behavior(In/Out) is controlled by the user
- **GUI (Graphical User Interface)**: allows users to communicate with electronics through an interface
- **Performance FFC**: Lepton Thermal Imager shutter that performs a flat-field correction
- **Servo**: motor that will be controlled by Pulse Width Modulation (PWM) that moves in an angular motion
- **I2C**: serial computer bus that is multi-master, multi-slave, and single ended used to connect lower speed peripherals to processors
- **PWM**: a digital signal that is used to control a circuit
- **GitHub**:web-based git repository hosting service
- **QT**: a cross-platform that can be written in your choice of language regardless of how many target platforms you will use

System Architecture (High-Level Picture)

      The Raspberry Pi 2 is a single board computer that does not include a built-in hard disk. Instead the Pi relies on a microSD card for booting and long term storage. The design includes a Broadcom BCM2836 SoC, with a Quad Core ARM7 900 MHz processor, VideoCore IV GPU, and 1 Gigabyte of RAM. The Pi uses a Linux-kernel based operating system, includes four USB 2.0 ports, an ethernet port, and the GPIO library. If it is unavailable to remotely connect to the Pi then we need a keyboard, mouse, HDMI cable, and a screen in order to be able to login. The Pi holds the information that controls the FLIR Lepton Camera and both servos.

      The FLIR Lepton is a longwave infrared (LWIR) imager sensor with a resolution of 80 (h) x 60 (v) pixels into a camera that is smaller than a dime and thermal capabilities. It is able to interact with mobile-device interfaces and other electronics. The Lepton has integrated digital thermal image processing functions, such as noise filters and non-uniformity correction. We will be using the Lepton to detect the temperature of a given image and return the coordinates of the maximum temperature found. The

temperatures are calibrated according to the surroundings. After retrieving the coordinate it is sent to the servos to move them to the correct x and y coordinates.
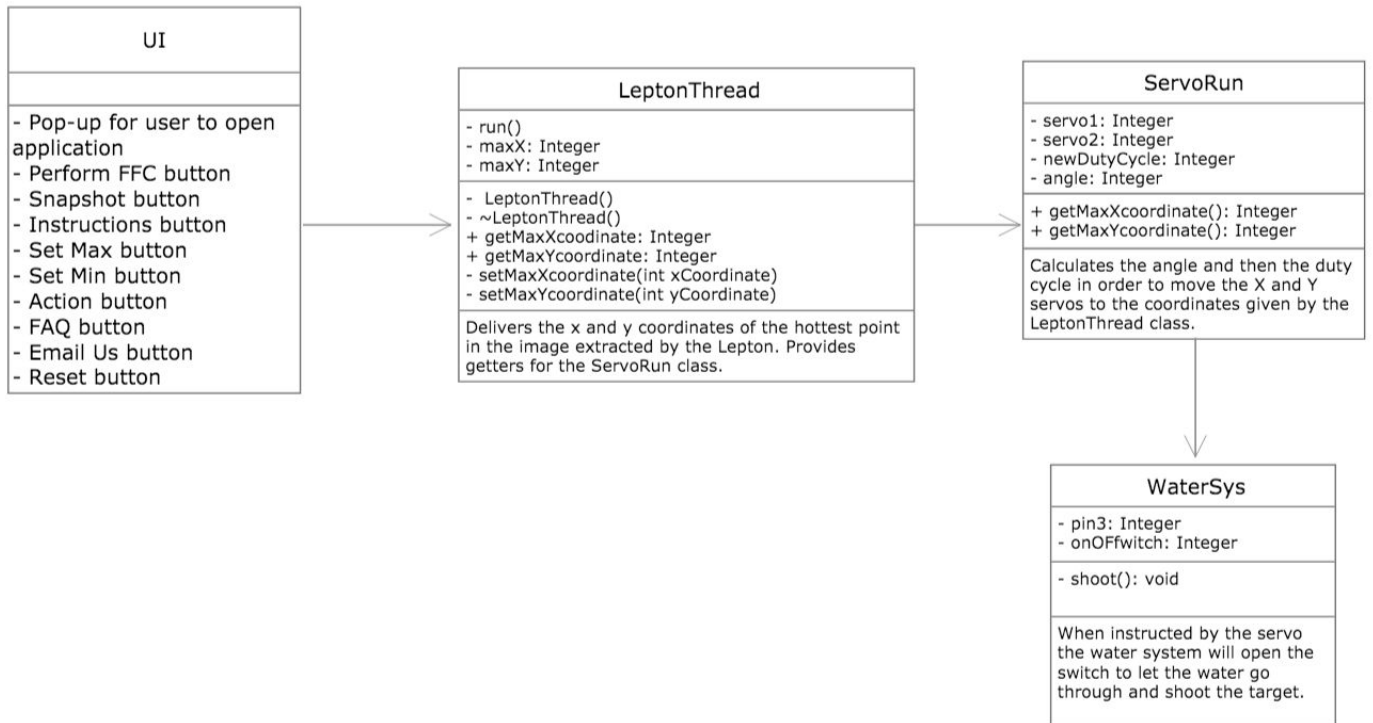
The project will include two A3020 Sub-Micro Digital High-Speed Metal Gear Aircraft Servo servos. One moves horizontally and the other vertically. The servos are controlled by the raspberry pi. Depending on the x and y coordinate the code finds the angle to which the servos must be moved to. Once there the tube that will shoot water to the detected hot object, located on top of the Y servo, is activated. The servo python file uses the GPIO library which allows mapping of provided pins to manage the servos.

The water system relies on a 12 volt motorcycle battery to give charge to the windscreen water pump which is connected to a watertight reservoir. The system also uses a pair of jumper leads that connect to the battery and the water pump.
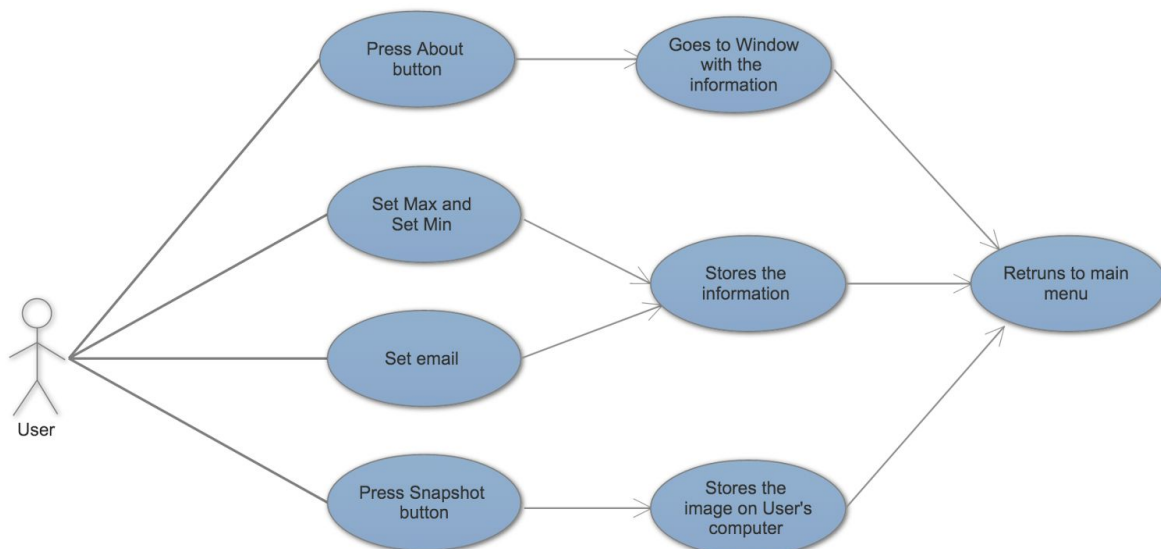
Most of the new features were added to the UI. When we run the file a popup box opens and asks if you wish to open ResQ Sensor. If you choose 'yes' then the application opens to the main window which includes the following buttons:

- Perform FCC: performs calibration by setting every pixel on the image to zero degrees.
- Snapshot: Takes an image with the lepton and saves the file to your computer.
- Instructions: This is to provide the user with a better understanding of how to interact with the UI.
- Set Max and Set Min: Allows the user to set the minimum and maximum temperature. If this temperature is reached then it will alert the user.
- Action: If the user chooses to allow the water pump to shoot water this button is pressed and the tube releases water.
- FAQ: This button will include potential questions the user may have, such as what each button does, information about the Lepton and Raspberry Pi, and contact information.
- Email Us: This is if the user desires to send us any questions, comments, or concerns.
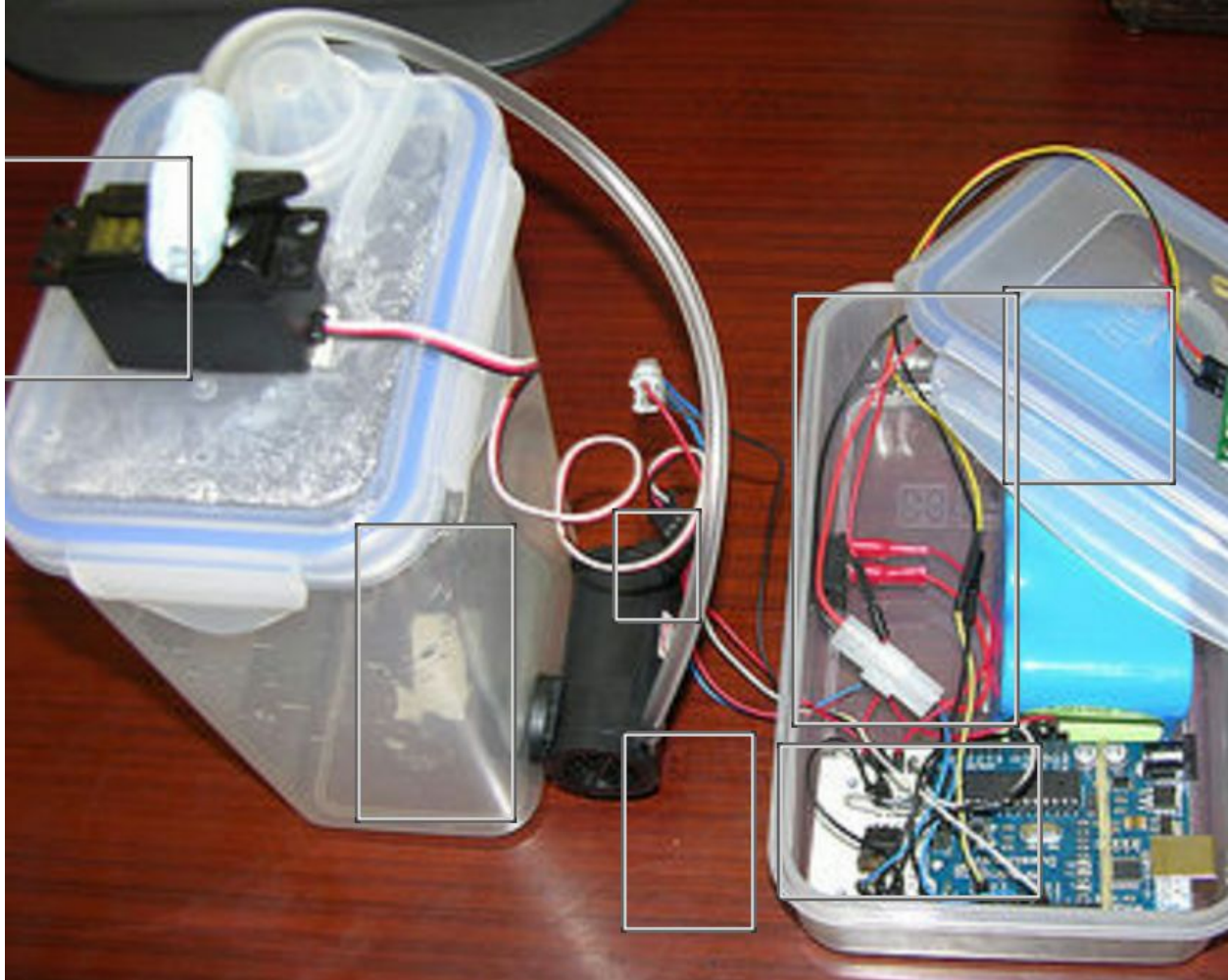
Classes, dependencies, and operations

UI

- Pop-up for user to open application
- Perform FFC button
- Snapshot button
- Instructions button
- Set Max button
- Set Min button
- Action button
- FAQ button
- Email Us button
- Reset button

LeptonThread

- run()
- maxX: Integer
- maxY: Integer

- LeptonThread()
- ~LeptonThread()
+ getMaxXcoodinate: Integer
+ getMaxYcoordinate: Integer
- setMaxXcoordinate(int xCoordinate)
- setMaxYcoordinate(int yCoordinate)

Delivers the x and y coordinates of the hottest point in the image extracted by the Lepton. Provides getters for the ServoRun class.

ServoRun

- servo1: Integer
- servo2: Integer
- newDutyCycle: Integer
- angle: Integer

+ getMaxXcoordinate(): Integer
+ getMaxYcoordinate(): Integer

Calculates the angle and then the duty cycle in order to move the X and Y servos to the coordinates given by the LeptonThread class.

WaterSys

- pin3: Integer
- onOFfwitch: Integer

- shoot(): void

When instructed by the servo the water system will open the switch to let the water go through and shoot the target.

Use Cases for the User on UI

User

- Press About button → Goes to Window with the information → Returns to main menu
- Set Max and Set Min → Stores the information → Returns to main menu
- Set email → Stores the information → Returns to main menu
- Press Snapshot button → Stores the image on User's computer → Returns to main menu

Similar Product (used as example)

Requirements:

<u>User Cases:</u>
1. Users can learn how to use ResQ Sensor in the About section on UI.
2. Users can click on the email us button on UI to send us an email.
3. Users can reset ResQ sensor by clicking the reset button on UI.
4. Users can turn on the ResQ sensor by plugging it in the outlet.
5. Users can only set a max temperature they want the room to be in.
6. Users can only set a min temperature they want the room to be in.
7. Users can set both min and max temperature at the time.

8. Users can set phone number for the alert.
9. Users can set email address for the alert.
10. Users can go to UI check on the room through the web camera.
11. Users can take a snapshot of the current image on ResQ sensor cam.
12. Users can check the camera angle through UI.
13. Users can set the static camera angle though UI.
14. Users can also set the camera to move its camera angle through time.
15. Users can fill the water tank with any amount of water that's suitable to the water tank.
16. The sensor will still be functional if the water tank is empty.
17. The sensor will send the user a notification if there is no water in the tank.
18. The sensor will send the user a notification if the camera is off.
19. The sensor will send the user a notification if the water got onto the ResQ sensor.

User Stories:
20. The sensor will detect the temperature change once the temperature in anywhere of the room changes (+/- 10F of the desired temperature).
21. The sensor will capture a image of area of temperature change
22. The sensor will send the captured image to the user through mobile app or web application.
23. Users check the image and send a signal to the Pi if they want the Pi to take actions.
24. the Pi will start timing after the image is sent.
25. If the Pi doesn't get a signal from the users within 5 minutes after the picture sent, it will take actions by default.
26. The Pi will automatically set the position of the pump through the use of the servos to aim at the target area.
27. The pump will splash water on the area of temperature change.
28. The Pi will send the user a notification once the temperature is back to normal.
29. Users can set the direction of the pump manually.
30. Users can click the button on UI to shoot the water without a temperature incident happening.
31. Users can turn off the ResQ sensor by unplugging it from the outlet.
32. Your house is safe and everyone is happy!

*note: the sensor means ResQ sensor, the device, not the sensor part of the system.

Prototyping code (under construction):

github URL:

Test Cases:

UI:

1. `bool testQbutton()`
   test if all the Qbuttons have callable callees
2. `bool testNotification()`
   test if the notification is sent
3. `bool testSnapShot()`
   test if the snapshot is stored somewhere
4. `bool testGUIFeature(fucntion *)`
   test if the individual function works properly
5. `bool testOpenReQ()`
6. `bool testCloseReQ()`
   test if the hardwares could be activated or deactivated by the UI

Hardware:

Servo Test:

7. `bool testMoveServorX()`
8. `bool testMoveServorY()`
   test if the moveServo functions works
9. `bool testGetServoX()`
10. `bool testGetServoY()`
    test if the servo position could be fetched

Lepton Test:

11. `bool testSceneX()`
12. `bool testSceneY()`
    test if the hottest point's coordinates are in record
13. `bool testLepton()`
    test if the lepton is sending stream to the raspberry pi

Water Pump Test:

14. `bool testPumpSwitch()`
    test if the water pump switch is functioning

Appendices

HARDWARE
- FLIR Lepton LWIR (80x60 pixel): (more specific on what lepton camera)
- Raspberry Pi 2
- A3020 Sub-Micro Digital High-Speed Metal Gear Aircraft Servo
- Water Spritzer:
  - Windshield Washer Pump with Grommet Fits BMW
  - SMC TUS Series Translucent Soft Polyurethane Tubing
  - Banjo NIP050-SH Polypropylene Pipe Fitting

SOFTWARE
- Text Editors:
  - Sublime
  - Vim
- Github
- Libraries:
  - GPIO
  - QT
  - I2C
- Programming Languages:
  - C++
  - Python