Department of Electrical Engineering
University of Washington

E E 474 Lab 5

Geophysical Detecting Tank & Video Recording

The report and its contents are solely our team's work and cites outside references for work that is not our own.

Signature:
Xiaoyi Ling, 1428502
Tiffany Luu, 1435225
Pezhman Khorasani, 1640743

TABLE OF CONTENTS

Introduction

The idea behind this lab was to create a tank that we could control remotely using Bluetooth, that could collect data about its surroundings and send that data back to the user. To implement this, we used the Beaglebone's GPIO pins, a Sparkfun Bluetooth module, 4 distance sensors, 4 LEDs, a photoresistor, a temperature sensor, a USB controller, and a H-bridge driver to make a bluetooth controllable machine. Using the joystick or D-pad on the controller, we could move the machine to forward, backward, left, and right, and also we could also stop the tank.We had 4 sensors for stopping the machine when it encountered an obstacle on either side.
Additionally, we added a photoresistor and a temperature sensor to the ADC inputs to read data from the environment. The data about the current temperature and brightness was printed to the computer with transferring data from machine to computer via bluetooth.
Despite not having it connected to the tank, we also used a webcam to examine the ability of the Beaglebone to capture, transcode and send live video streams and output video streams to a video file.

Design Specification

Geophysical Detecting Tank with Bluetooth Control

User Interface: The user will be using a controller to control the tank. They will send these controls through a bluetooth terminal. The user will see the information from the sensors on this terminal as well. The tank's function will start upon bootup. The individual functions of the tank are described below.

Tank Drive Mode

Input: Controller instructions received by Bluetooth, distance sensor readings
Output: The tank will drive under instructions received from controller through Bluetooth
Error check: The program is able to check if a file is opened successfully. If not, the system will print error message to the terminal and exit

This program will run the tank when started, according to user input, and avoid obstacles according to sensor input.

- When the Beaglebone is powered on (via the power bank), the script to run the programs will run on boot up.
- A controller is connected to the terminal window that send instructions to Bluetooth. When pressing button on the controller, different instructions will be sent to Bluetooth through terminal window.
- The tank will stay still upon powered on waiting for instructions. A program will read from the Bluetooth and change the direction accordingly. In our case, a letter "w" means move forwards, a letter "s" means move back**,** a letter "a" means turn left**,** a letter "d" means turn right**.**
- The distance sensor program will update the drive program if the sensor values change.

- This process defers to the distance sensor readings. If the tank is moving and it gets a sensor reading in that direction, it will stop. If the user attempts to move the tank but there is an obstacle in that direction, the tank will not move.

Detecting Temperature and Brightness

Input: Temperature and Brightness from surrounding environment
Output: Information sent to computer console using Bluetooth
Error check: The program is able to check if a file is opened successfully. If not, the system will print error message to the terminal and exit

This program will read inputs from a temperature sensor and a photoresistor. It will then convert the temperature data from mV to Fahrenheit. It will convert the brightness data to an overall percentage. It will then send this data to the computer console via bluetooth.

- After getting an appropriate input, the program will convert the data from the sensors to the corresponding temperature and brightness values.
- The program will send the data for temperature and brightness to the computer console which in our case is Coolterm via bluetooth. On default, the monitor will display the temperature and then unit "°F" and brightness: " followed by the percentage and "%".

Avoiding Obstacles Using Distance Sensors

Input: Distance from obstacles
Output: Sends sensor data to the driving program, sensor status displayed on LEDs
Error check: The program is able to check if a file is opened successfully. If not, the system will print error message to the terminal and exit

This program will read the input from all four distance sensors and interrupt the drive program if the status of the sensors changes.

- The ADC reading is an input voltage from 0-1800 mV.
- The max reading from the distance sensor will be 1600.
- The default is set to average 50 samples, each sampled per 1 ms
- During the tank moving, four distance sensors are attached to the four sides of the tank to sense the distance to any object from each direction. When there are obstacles on one side of the tank, the sensor will update its status to indicate this, and a corresponding LED will light up.
- The status of the sensors has top priority. If the tank is moving under the instruction from controller, the tank will stop if the sensor indicates there is an obstacle in that direction, even if the controller is giving other instructions.

Video Recording and Streaming

Input: data captured from webcam Logitech 310
Output: Live video streaming in VCL media player or webpage 192.168.7.2:xxxx and video recording in .avi form.

This part examined three different possible ways for the tank to use webcam for video streaming and video recording using Beaglebone.

1. Video streaming using VLC media player. Change the IP address to the the address to receive output video in bash script Scan. Then run the script Scan and until it begins to encoding data. Terminate the program and copy everything shows on the terminal up to "Press ctrl-C to stop encoding" and paste them into a session description protocol file. In our case, we called it Video.sdp. Send the .sdp file to the computer that displays video. Then run the script Scan again and open .sdp file using VLC. Then the live streams captured by webcam will display in the VLC video player

2. Video streaming using webpage. This program is adapted from a precoded program called mjpg-streamer. It gets the mjpeg data from V4L2 and send it through a HTTP session. Make the file and run it with bash script Run. Then open the webpage at 192.168.7.2:xxxx, where xxxx is the port assigned for the video. In our case we used 8080. Then the webpage with address 192.168.7.2:8080 will be set up like Figure 1. Click on Stream button, the live video will be displayed.

Figure 1: Sample webpage for mjpg-streamer program

3. Video recording using avconv package. This program examines the video recording function using avconv package and beaglebone. Run the bash script Record and it should begin to record the video. Use ctrl-C to terminate the program and a new file output.avi will be generated in the Beaglebone. Send file from Beaglebone to the computer and then the video can be played. This method is adapted from Derek Molloy in his tutorial "Beaglebone: Video Capture and Image Processing on Embedded Linux using OpenCV"

Hardware

In order to fit all of the sensors, drivers, and LEDs to the beaglebone, we used an additional breadboard and mounted it above the one provided with the tank. A picture of this setup can be shown below in Figure 2. Note that the LED display is also on the top level; initially, we were sending the temperature and light reading to the LED display via shift register before rerouting it to the user via bluetooth, but the LED display is not used in the final project.

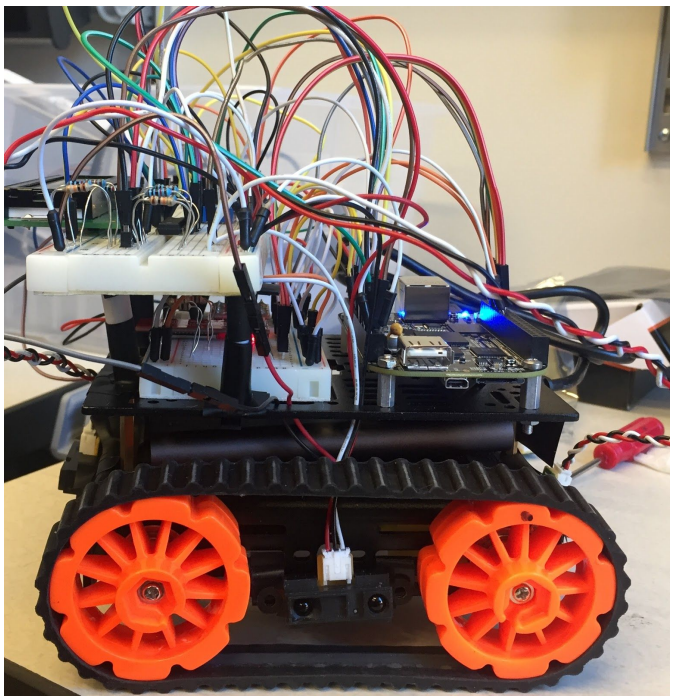

Figure 2. Overall setup

H-bridge Motor Driver

(The function of the H-bridge is the same as Lab 4)
An H-bridge motor driver (model TB6612FNG) was used to control the motors attached to the wheels of the tank. Input/output A was connected to the left motor, while input/output B was connected to the right motor. The PWM for each motor is used to control the speed the motor turns. The smaller the period/duty, the faster the motor turns. The wheels will turn in either direction depending on whether IN1 or IN2 is high. The connections for the H-bridge are shown in Figure 1 and Table 1 below.

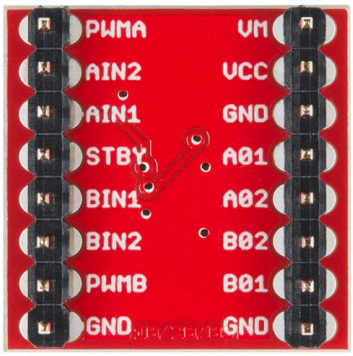

Figure 3: Image of the pin connections for the h-bridge driver

Table 1: H-bridge Pin Connections

H-bridge	Connections	Function
PWMA	EHRPWM1A	chA PWM input, controls the speed of motor A
AIN2	GPIO_66	chA input2, instruction pin for motor A
AIN1	GPIO_69	chA input1, instruction pin for motor A
STBY	Switch	No output to the motor when standby is low
BIN1	GPIO_47	chB input1, instruction pin for motor B
BIN2	GPIO_27	chB input2, instruction pin for motor B
PWMB	EHRPWM2A	chB PWM input, controls the speed of motor B
GND	GND	Power: GND (0 V)
VM	Battery pack	Power: ~6.5 V, power for the motor
VCC	SYS 5V	Power: 5 V, power for the H-bridge
GND	GND	Power: GND (0 V)
A01	(red) Left Motor	chA output1, output to motor A
A02	(black) Left Motor	chA output2, output to motor A
B02	(black) Right Motor	chB output1, output to motor B
B01	(red) Right Motor	chB output2, output to motor B
GND	GND	Power: GND (0 V)

For the standby pin, we connected an SPDT switch where the middle was connected to STBY, and the other two pins were connected to 0 V and 3.3 V. The logic table for the input pins to control the motor movement is show below in Table 2. The forward/reverse can be interchangeable depending on how the output is hooked up to the motor.

Table 2: H-bridge Controls

IN1	IN2	Behavior
0	0	Free
0	1	Forward
1	0	Reverse
1	1	Brake

Distance Sensors

We connected four distance sensors to each side of the tank. The distance sensor has a max output of ~3.3 V at around 10 cm. Objects past ~75 cm gave a 0 V reading. An image of this is shown below in Figure 2.

Figure 4: Distance sensor.

Because the max output of the distance sensor is greater than the minimum input of the ADC pins of the Beaglebone, we used a voltage divider to divide the output in half. The voltage divider consisted of two 10 kΩ resistors. The power was connected to SYS 5 V, the ground line connected to ground, and the output of the sensor connected to the power of the voltage divider. The max output of the voltage divider is ~1.6 V.

The output from the voltage divider is sent as input to the Beaglebone via its ADC pins. For the detection program, four LEDs will light when their respective sensors sense something. These LEDs are in series with a 1 kΩ resistor. The connections for the sensors and the LEDs are described below in Table 3.

Table 3: Distance Sensor System Connections

Beaglebone	Connection	Function
SYS 5V	Breadboard +	Power: V_{CC} = 5 V
DGND	Breadboard -	Power: V_{DD} = 0 V
GPIO_44	Green LED	Light up LED when signal from front sensor is received
GPIO_68	Red LED	Light up LED when signal from back sensor is received
GPIO_67	Yellow LED	Light up LED when signal from right sensor is received
GPIO_26	Blue LED	Light up LED when signal from left sensor is received
AIN0	Voltage divider for front sensor	Read the input voltage from front sensor after it is divided to half by a voltage divider
AIN1	Voltage divider for back sensor	Read the input voltage from back sensor after it is divided to half by a voltage divider
AIN2	Voltage divider for right sensor	Read the input voltage from right sensor after it is divided to half by a voltage divider
AIN3	Voltage divider for left sensor	Read the input voltage from left sensor after it is divided to half by a voltage divider

Bluetooth

The following Bluetooth chip in Figure 5 below was used for transmitting data from the user (via a laptop) to the Beaglebone. The bluetooth sends data to the Beaglebone via a UART. The baud rate used for our purposes was 9600 b/s. The connections for the bluetooth module are described in Table 4 below.

Figure 5: Sparkfun Bluetooth Module

Table 4: Bluetooth Module Connections

Bluetooth Module	Beaglebone	Function
GND	GND	Power: 0 V (Ground)
VCC	SYS 5V	Power: 5 V
TX	UART4_RXD pin 11	Transfer data to Beaglebone
RX	UART4_TXD pin 13	Receive data from Beaglebone
RTS-O	Not connected	Request to send
CTS-I	Not connected	Clear to send

Temperature and Light Sensor

The sensors that we used to get environment readings were the LM35 temperature sensor and the 200 kΩ photoresistor, shown in Figures 6 and 7 respectively.

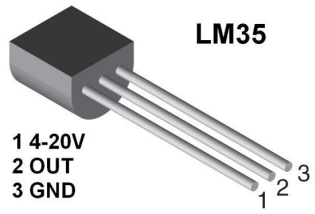

Figure 6: LM35 temperature sensor

Figure 7: 200 kΩ photoresistor

The analog input pins AIN6 and AIN4 are used for this part. The light sensor is in series with a 10kΩ resistor, and AIN4 is connected to the light sensor output across the 10kΩ resistor. The AIN6 is connected to the temperature sensor output (pin 2). A 10kΩ resistor was connected in parallel across the output and GND to stabilize the output. The model of the temperature sensor used is LM35. This connections for both sensors are shown below in Tables 5 and 6.

Table 5: LM35 Connections

LM35	Beaglebone	Function
V_{SS}	SYS 5	Power: V_{CC} = 5 V
Out	AIN4	Read input voltage from temperature sensor. For voltage, the value is displayed as millivolts. Output range: 0 - 1 V
GND	~0.05 V	Power: V_{SS} = 0 V

Table 6: Photoresistor Connections

Photoresistor	Beaglebone	Function
Photoresistor/pin1	VDD_ADC	Power: V_{DD} = 1.8 V
Presistor pin2/ & 10kΩ resistor/pin1	AIN6	Read input voltage from photoresistor. For voltage, the value is displayed as millivolts. Output range: 0 - 1.8 V
10kΩ resistor/pin2	GND	Power: V_{SS} = 0 V

Controller

The Logitech F310 USB controller was used to control the tank. We plugged the controller into a laptop and used the controller to send data to the Bluetooth. The back of the controller has a switch between X-input mode and D-input mode. We set the controller in D-input mode so we could program the controls to keyboard inputs. We programmed the D-pad and left joystick to the WASD keys, and the blue "X" button as a brake.

Figure 8: Logitech F310 controller for controlling tank

Camera

In this lab we used Logitech C310 for video capturing, streaming and recording. The USB port of the camera is connected to the USB port of Beaglebone.

Figure 9: Logitech C310 camera for video capturing

Software

The following is a short description of the user interface on the laptop that sends data to the bluetooth, and the C files and programs that are used. The C programs are used to move the tank, to send signals from the distance sensors to the drive program, send signals from the bluetooth to the drive program, and send data about the environment sensors to the user via bluetooth. There is also a description of the programs/scripts used to record and stream video.

User Interface

We used the CoolTerm terminal to send data to the Bluetooth module. The terminal settings were set to a baud rate of 9600, 8 bits, no parity, and a stop bit. In order for the Beaglebone side to register a read, you needed to enter a newline character first.

For the controller, we set it to Direct Input mode. Plugging the controller in D-Input will bring up the following window as displayed below in Figure 10.

Figure 10. Logitech Profiler for using the controller in D-Input mode.

D-Input Mode requires you to program the output for each button for a particular game. Instead of programming the output to a game, we programmed the output to the CoolTerm terminal. You can see some of the reprogrammed commands in Figure 10. The left joystick axes were reprogrammed to “Letter AD” and “Letter WS”, while the D-Pad on the left top was reprogrammed to “Letter Movement.” This is done by recording what output you want, and

setting it to an appropriate movement. An example of how the letter presses were programmed can be seen below in Figure 11.

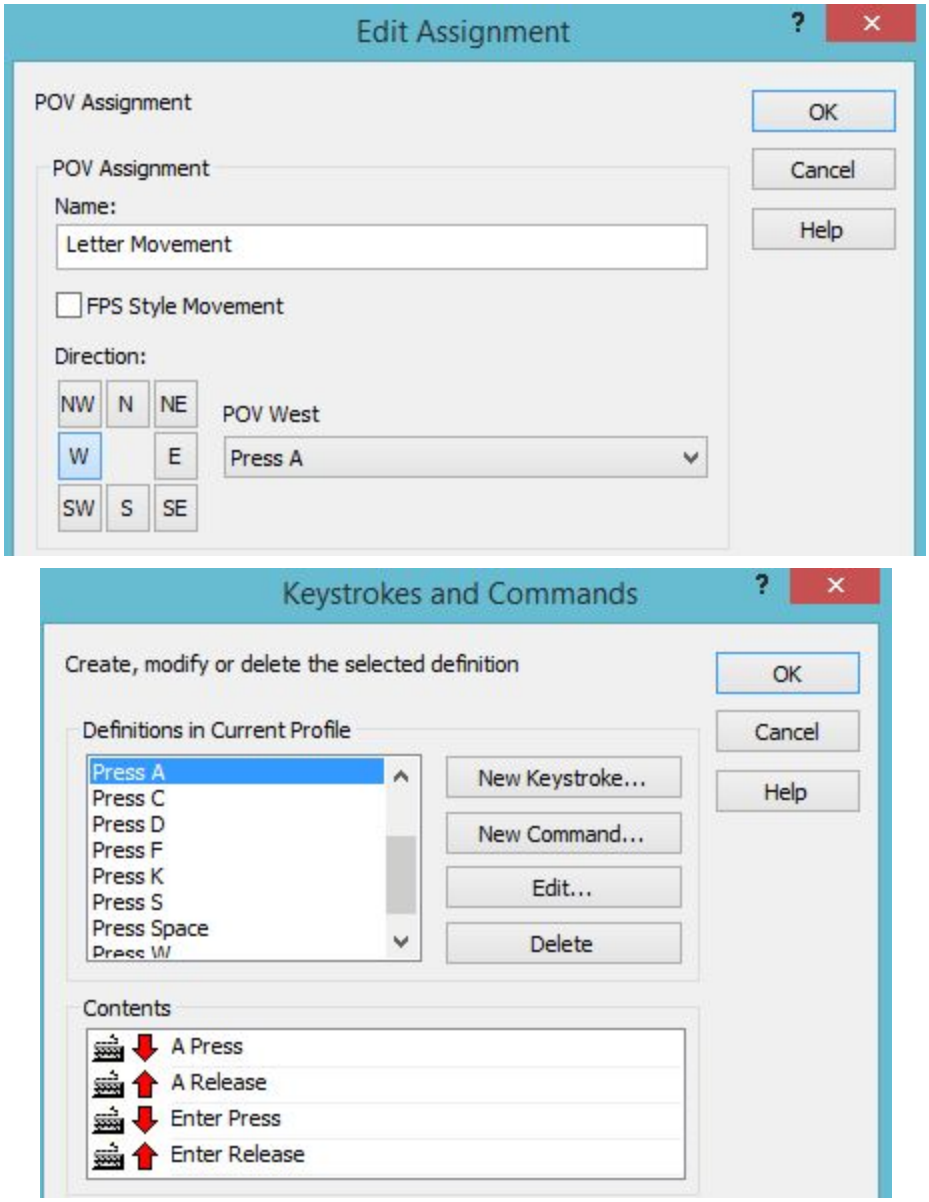

Figure 11. Programming the Keyboard commands for the terminal.

Aside from the D-Pad, the zones of the joysticks can also be reprogrammed to detect different areas, although we found that the accuracy of the joystick was not great. The "X" key was programmed to space, which brakes the car. For user convenience, the right joystick could be used like a computer mouse, while the left and right back triggers could be used like a mouse left-click and right-click respectively.

Geophysical Detecting Tank with Bluetooth Control

startup.sh: Script that runs when the Beaglebone is booted up. We used a startup service via systemd to execute this script when the Beaglebone turns on. This script calls the drive code first, and then gives that process ID to the distance sensor and bluetooth receiver code as one of their arguments. It assumes that all of the necessary files have been precompiled before execution.

btreceive.c: Receives input from the Bluetooth module via the Beaglebone's UART file. The Bluetooth/UART baud rate was set to 9600, and the protocol was 8 bits, with the parity off. This program receives data from the user after the user hits the newline character in the terminal. If the program receives a "w", "a", "s", "d", or " " character, it will send the corresponding signal to the drive program to get the tank to change directions.

dsense.c: Reads input from four AIN pins of Beaglebone that are connected to the distance sensors every 1 ms. It computes the average value every 50 samples. It will then check the average value in the order of front, back, left and right. If any value is larger than 900, it will update the corresponding character in a character array with a '1'. If the value is less than 900, it will update the array with a '0'. The program saves a current and previous reading; if the current reading is different from the previous reading, then it will send a signal to the drive program. The program will then send the current reading array through a pipe to update the drive program.

drive.c: Receives signals from the bluetooth and distance sensor programs, and uses these signals to decide the behavior of the tank. This program uses PWM and GPIO pins to control the behavior of an H-bridge driver, which controls the wheels of the tank. If it receives a signal from the distance sensor program, it will update its distance sensor reading values. If the value is "1", or a sensor is close to something, it will light an LED. If the tank receives a "w", "a", "s", "d", or " " signal from the bluetooth program, the tank will move forward, turn left, move backward, turn right, or brake, provided that the sensor does not sense anything in that direction. If the tank is already moving in a direction and the sensor senses something, it will stop.

sensor.c: This program reads inputs from the ADC pins of the beaglebone corresponding to the temperature sensor and light sensor. It will send the data collected from the sensors to the user via bluetooth every 3 seconds (the polling time is adjustable).

Makefile: Compiles the programs above with the right executable names for the startup script. You can also run the startup script using a make target.

(Extra) basicreq.c: Contains only the basic requirements of this lab, which was to use a bluetooth to control the tank. This only has the tank following the user controls from the bluetooth, also using WASD.

Video Recording and Streaming

avconv: all the three methods to record and stream video use avconv package. This is a package released from Libav which is used to capture and convert media data to different forms and send them to specific addresses.

Video streaming using VLC media player

This program uses user datagram protocol to set up connection between Beaglebone and a computer by avconv package. Script Scan should stay in the Beaglebone to receive data from webcam while file Video.sdp should be sent to the computer to play live video stream using VCL media player.

Scan: a bash script using avconv to capture video stream and send it to the IP address specified by the user. It pairs with Video.sdp to perform live video streaming on the computer using VCL media player. -f video4linux2 -input_format mjpeg" tells device driver video4linux2 to capture input in format mjpeg; "-i /dev/video0" takes the input from the pipe /dev/video0; "-vcodec h263p" transcodes the stream to h263p; "-r" sets the number of frames per image of the output stream; "-f rtp rtp://192.168.142.140:1234/" Force rtp to output to address of my PC on port 1234

Video.sdp: a session description protocol file. receives data sent from Beaglebone by Scan and play them using VCL media Player.

Video streaming using webpage

This program uses a premade project called media-streamer to capture live video and send it to a IP address specified by the user using avconv

Folder media-streamer: program adapted from mjpg-streamer program on the Internet. Use Makefile to make the executable program mjpg-streamer and run it using script Run.

Video recording

This program uses avconv to capture video and convert it to an output file in .avi format

Record: a bash script that uses avconv to record the video captured by the camera and store in the output.avi.”-f video4linux2 -input_format mjpeg" tells device driver video4linux2 to capture input in format mjpeg; "-i /dev/video0" takes the input from the pipe /dev/video0; "-vcodec h263p" transcodes the stream to h263p; "-r" sets the number of frames per image of the output stream; "output.avi" sets output file to be output.avi.

Testing

Tank Hardware

There were multiple issues with getting the wheels to turn. The torque provided by the motors of the tank were not enough to turn the tank wheels efficiently with the treads on. Some of the nails for the wheels were also too tight, so we loosened the wheels to get them to turn better. However, we could not manage to get the wheels to match the speed perfectly on both sides, so driving forwards over a long distance, you will see the tank start turning left.

The tank's turning was therefore extremely ineffective. We tried three methods of turning the tank. For these examples we will describe the tank turning right. To turn right, we tried turning the left wheels forward fast, and the right wheels very slowly. This resulted in an extremely slow turn - the tank would move forward quite a bit before turning. We also tried rotating the tank, by turning the left wheels forwards and the right wheels backwards. The stiffness of the treads seemed to prevent the tank from moving at all with this method. Finally, we attempted to just spin the left wheel very quickly. This causes the tank to rotate very slowly. We decided to go with this method, although it was still ineffective, because it provided the sharpest turn (albeit very slowly).

Beaglebone Crashes

A couple hours before the demo, the Beaglebone crashed. Something with this crash seemed to mess with the program execution and the Bluetooth connectivity. Despite not running any programs, the user terminal would receive junk from the Beaglebone and the scripts would start the wrong program/send the wrong commands. We fixed this by switching out the Beaglebone, but it was suddenly harder to connect to the Bluetooth module from the user side. Because of this, we decided to send the temperature and brightness data to the user only, and let it run on the default of Fahrenheit rather than prompting the user for temperature conversion like we had planned. We also disconnected the LCD screen, which we were also going to send the temperature/brightness data to as well.

Video Streaming and Capturing

Our previous idea was to use WiFi to send video streams and display them on the computer. We used a WiFi adapter and managed to get WiFi set up in my apartment. However, the WiFi provided by University of Washington needs login to have access and the adapter cannot catch the signal. We also tried to turn one computer into a Hotspot to provide WiFi, but it seemed that the adapter did not recognize the virtual router. So we could not set up Wifi in the lab and this plan was aborted. We then decided to record the surrounding environment using a webcam and send the video from Beaglebone to computer when we reconnect beaglebone to computer. However, the system crashed before we tried this and we were worried that adding this program would cause crashes again, so we separated the programs about camera from the driving of the tank.

Besides what we mentioned above, we also had many difficulties on running avconv. The program kept giving us “invalid output detected” error and we finally fixed it by changing many times for formats, resolutions and pixel rates until the error disappeared. But we were not able to fix the problem thoroughly. The video display using VCL had about 1 second delay and occasionally, the output video produced by Recond had super long playing time. It will run about 2 hours for about only 20 seconds capturing time. We were not able to figure out the reason but we believed that some data were wrongly transcoded when the program was running.

Conclusion

Throughout this lab, we learned to create a bluetooth connection for our machine by writing and implementing UART. We also learned to use a sensor by ADC programming with C code. Additionally, we practiced using the AIN pin by reading data from temperature sensor and photoresistor, and combining that function with commands to the monitor. Our program can be improved by adding live video streaming or video recording feature but they will slow down the system running on the Beaglebone and potentially caused crashes. Overall, our project is a success in terms of following the instructions from controller, avoiding obstacles and detecting temperature and brightness of surrounding environment, although the crashes and functionality could be improved.

Reference

Molloy, D. [DerekMolloyDCU]. (2013, May, 25). Beaglebone: Video Capture and Image Processing on Embedded Linux using OpenCV [Video file]. Retrieved from http://www.youtube.com/watch?v=8QouvYMfmQo

Code for mjpg-streamer adapted from: https://github.com/shrkey/mjpg-streamer