




Machine Problem 2

DOCUMENTATION

CELINE ROSE D. MARCELO

2014-15480



INTRODUCTION

Being part of the second half of the EEE 13 course, Machine Problem 2 includes the concepts of socket programming and signals. Using TCP, this project aims to establish communication across multiple clients. This idea is simulated by a number of sensors sending data to a server.

This documentation will provide an overview of what each C file does. Detailed explanations are written in the form of inline comments.

IMPLEMENTATION

Server

Usage

```
./server port -l/-g threshold
```

Overview

The server side of this MP was taken from ME8, which focuses on sockets. The same method, `select()`, was used to handle multiple clients simultaneously.

The code was roughly based on an example from Beej's Guide to Network Programming (<http://beej.us/guide/bgnet/output/html/singlepage/bgnet.html>).

The sensors' information (data, PID, sensor number and file descriptor) are stored in an array of structs. This was declared globally to make way for convenient manipulation by the functions. The program also forks to execute the plotting module simultaneously while the server is running.

Defined Functions

Function	
<code>void kill_all (void)</code>	Signal handler for SIGINT. Kills all clients before ending server program.
<code>void plot (int index)</code>	Plots contents of appropriate sensor log. Is actually an external program executed from within the server using system().
<code>void write_log (int index)</code>	Writes sensor data to log file, which will be read by plot().
<code>void check_data (int index, char *mesg, char *g_l, float threshold)</code>	Checks if received data falls within threshold, and terminates client involved. Prints data otherwise.

Client

Usage

```
./client IP:port -s sensornum -l/-g threshold
```

Overview

To simulate sensors sending data, the client side of the Machine Problem makes use of the classic send() function. This program was implemented so that the client initially sends its sensor number and PID first to make itself known to the server. Normal sending of data (read from the appropriate .txt file) is initiated afterwards.

The code's framework is from <http://www.programminglogic.com/example-of-client-server-program-in-c-using-sockets-and-tcp/>.

Defined Functions

Function	
<code>void extract_info (char *ip_port, char *ip, char *port)</code>	Extracts the IP and port of server from program arguments.
<code>void send_data (int clientSocket, char *g_l, float threshold, FILE *fp)</code>	Checks if data falls within threshold before sending to server. Has a required delay of 1 second between iterations.

Plotting (Bonus)

Usage

```
./plot sensornum
```

Overview

Using PLplot's C language binding, this program plots the data from the appropriate sensor logs. This is executed after a client is either terminated or was disconnected. It can be used as a child process of the server, or run independently. The plot is automatically saved to an SVG file.

The code's skeleton can be found in http://www.physics.csbsju.edu/~jcrumley/222/examples/plplot_example.C.

PLplot was easy to use and understand, with only minimal configuration (e.g. extreme values, colors).

PROBLEMS ENCOUNTERED

Since the bulk of the Machine Problem was based on a previous Machine Exercise, the journey to its end was swift.

However, the developer had a hard time at the beginning when figuring out how to pass the sensor's number to the server. Also, due to the developer's inexperience in external libraries, a few hours were spent on searching for an easy-to-understand method of plotting the data.

All in all, this Machine Problem was fun. It is a magnificent way of ending the course. 😊