ENGTeam 6

James Nguyen
Jonathan Russo
Joseph Alagon

Appendix A

Milestone Description	Tasks that needed to be completed	Responsible team members	Date of completion
Solar Collection GUI Planning	Determining what components the GUI will feature such as date and approximate location edit boxes	James Nguyen Joseph Alagon	03/13/2013
Solar Collection GUI Implementation and Development	Actually code the GUI with object oriented programming GUI editor	James Nguyen Joseph Alagon	03/ 14/2013
Plot Interface GUI Design	Determining what components the Plot Interface GUI will	Jonathan Russo	03/15/2013
Plot Interface GUI Implementation and Development	Constructing Code for the GUI	Jonathan Russo	03/16/2013
Debugging of Solar Collection GUI	Run and Confirm the Solar works. Fix all errors with code.	All	03/17/2013
Debugging of Plot Interface GUI	Debug Plot Interface GUI	James Nguyen Jonathan Russo	03/17/2013
Solar Survey Data Gathering	Gather solar data from 28 locations on UC Davis Campus	All	03/16/2013
Video Production	Make and edit video for the report	Joseph Alagon	03/17/2013
Group Report	Write up group report	Joseph Alagon	03/18/2013
Beta Testing of GUI	Beta Testing of both GUIs	All	03/18/2013

Jonathan Russo

For this project my main responsibility was Task 2. I also wrote the code that ran the different sweeps that the Arduino module took. For the Arduino code I manipulated the existing Lab 7 code. I created the loops that run the open and load sweep, and I refined them, making them run faster and more efficient by preallocating variables, decreasing the pause time, and setting the waitforbuttonpress function that allows the user to pause and remove the open circuit jumper before continuing with the next loop. I was also in charge of task 2, the plotting user interface and interpolation. For this I created a GUI with mutually exclusive button groups that allows the user to choose between the types of data to plot. I also allowed the user to plot the data using contour plots along with interpolated data where it was not collected by the user. The data is scaled by a specific code block that scales the colors of the contour plot individually to each type of data being inputted. This is what makes our code different and really allows the user to view small amounts of variation. If the data were scaled by 255, one may not see the differences if the data does not vary much (a difficult to uncover lesson I learned debugging the plots). I wrote the entirety of task 2 as well as the code that runs the Arduino sweeps. I also helped James with the debugging of his code and provided insight, as he did for me too.

James Nguyen

The function I played within the group was to construct the graphic user interface that is enables the user to collect sunlight. The other role I had was assisting in the debugging of the other interface used to interpolate and plot the solar data.

The interface for collecting the sunlight includes functions that are not our own work. These files that contain the functions are the a*rduino.m*, *ginput.m,* and *getAvailableComPort.m*. The ginput2 function is what allows one to zoom in, zoom out and select their location at the same time. Originally, the function is created by Carlos Adrian Vargas Aguilera as indicated by this link: https://www.mathworks.com/matlabcentral/fileexchange/20645-ginput2-m-v3-1-nov-2009. The reason why ginput2 is useful is because while using the standard ginput, the user cannot zoom in or out without first inputting their coordinates. The arduino function is taken from Lab 7 of the Engineering 6 Labs. The main use of this function deals with the connection between the computer and the Arduino module. Lastly, the getAvailableComPort function allows the user to find out which serial port their Arduino is connected to more easily than the instructions that Lab 7 provide. This function is written by Daniel Lavoie as indicated by the link: http://www.mathworks.com/matlabcentral/fileexchange/9251-get-available-com-port.

I was also involved in the debugging and the development of the Plot Interface GUI which interpolates and plots the contour plots for the user data. The major problems with this devlopment process was getting the data to work with the function TriScatteredInterp. The main reason for this issue was due to

the data not being scaled properly. My main contribution to this part was locating syntax errors and constructing the algorithm to plot the Max Power of the raw data and the Average Voltage of the raw data. In conclusion, my role dealt majorly with debugging and GUI development.

Joseph Alagon

My main role in the group was to help plan out the solar gathering GUI, test it in the field by collecting solar data with the GUI, and helping make changes to it to make solar gathering more user friendly. My other role was collecting the 18 data points (28 in our case), making the YouTube video, and writing the final report.

In regards to my participation with the solar gathering GUI, James and I planned out what the GUI was originally going to have in regards to interface, and I worked with James in writing the code and creating the class in which the user inputted information and data collected from the Arduino module would be saved in. The main problem we faced when making the solar gathering GUI was trying to figure out a way to save the classes when the user was done collecting data at a certain location. We fixed that by making the global class "ALL_TRIALS" to store each individual location by telling the user to save his/her data in a trial number.

When we finished a version of the GUI, I would use the GUI and gather data using it, and worked with James with debugging errors in the code, and making the GUI more user friendly. Once we finished the final version of the solar gathering GUI, I used the GUI to collect solar data at 28 locations all over Davis. And lastly, outside of the programming, I also filmed/edited the YouTube video for the group, and wrote the final report.

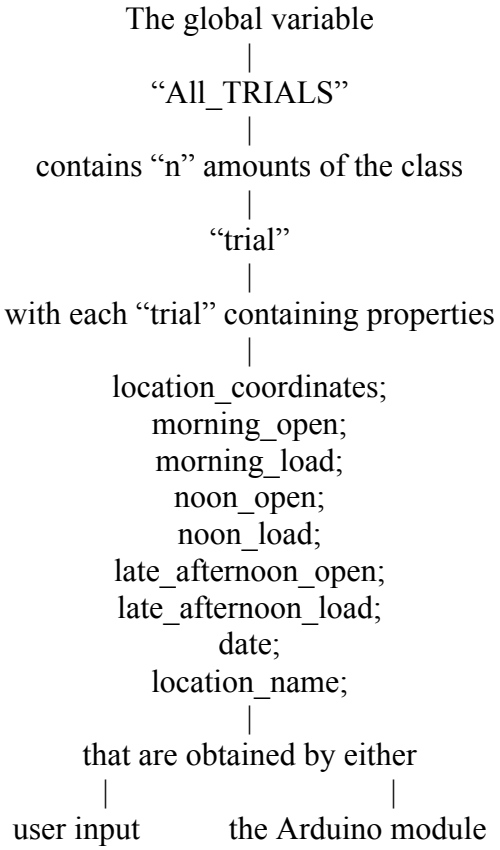
"All team members have read the task summaries contained in this report and have been given an opportunity to comment. "

This report will explain the program in two parts: The first part will talk about how the solar gathering GUI works, and the second half will talk about how we plot the data through the solar plots GUI.

When you first use the Solar_Gather_GUI, first we ask the user to input information such as time of day, date, location, and etc. Next, the user has to define what time of day they are recording the data. Afterwards, the user is prompted to collect solar data through the Arduino module by clicking the "Collect sunlight" button. Once the user is done collecting his/her data, the user is prompted to set a trial number as a way to save his/her inputs and data. So from here, two questions came up when making the GUI: how is the user inputted information and the solar data gathered from the Arduino module saved, and when the user starts a new trial (when I say new trial, I mean a new trial number with new inputs and solar data, not restarting the program entirely), how is the previous trial saved without that data being lost or overwritten by the new trial?

To answer the first question, we used a technique called "Object Oriented Programming." All of the user inputs and the solar data gathered from the Arduino Module are saved into properties that are part of a class called "trial". In regards to the serial port, we made a global variable called "COM_STRING" that will get the user inputted serial port, and set it equal to the "arduino.obj", so that the load and open circuit sweep can work when the user presses the "Collect sunlight" button. In regards to the map of UC Davis, we used a function called "ginput2" to store the x and y coordinates that come in the form of the user input of a right click on the map of UC Davis. For date and approximate location, we just used the get function and stored the user input as strings. Not sure about how trial number works.

To answer the second question, we used another global variable called "ALL_TRIALS". The variable "ALL_TRIALS" stores each trial into a 1:n cell structure, where n is the number of trial numbers that the user has saved. For example, if the user has done three different trials and saved each trial using trial numbers 1-3, then the variable "ALL_TRIALS" will be a 1x3 cell structure, with each cell structure containing all the properties in the class "trial" saved by the user during each trial. The flow of the first GUI works like this (next page):

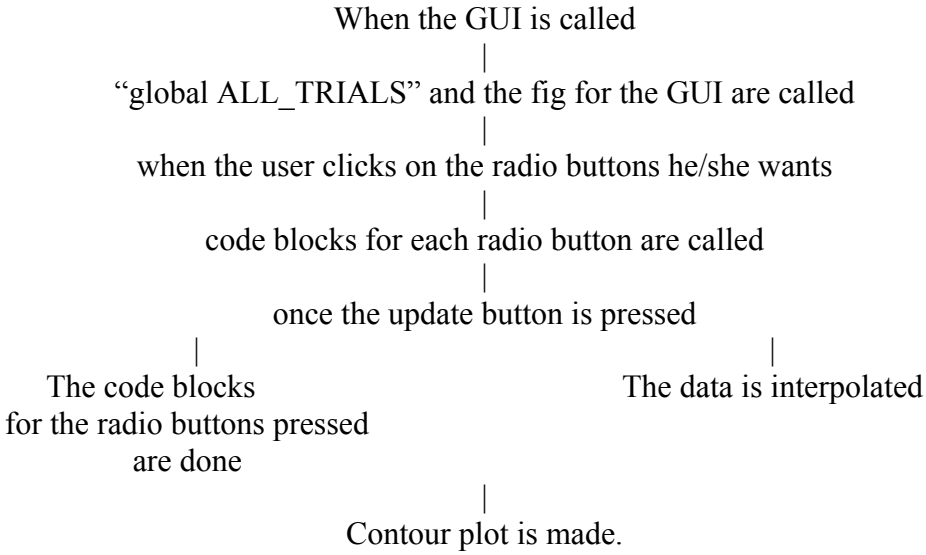


After the user has finished collecting solar data at "n" amount of locations, they are to press the "Plot Interface" button. This brings up a new GUI that is called "GUI_Plots". In this GUI, the user picks what combination of variables they want to see in a contour plot of the campus (Raw vs. Max Power, Average vs. Voltage, etc.). After the user clicks the update button, a contour plot the user chooses pops up. So from here, three questions arise: how does the second GUI call the data from the previous GUI? How does the plotting GUI know what combination of variables to graph for the contour plot? How do we interpolate data at locations where data is not available?

To answer the first question, we can just call the global variable "ALL_TRIALS". Because the first GUI calls the second GUI to pop up with a button click, the data saved in "ALL_TRIALS" is still available for use, so just calling "global ALL_TRIALS" is sufficient in calling the data saved from the first GUI.

To answer the second question, this can be solved by a series of if-else statements. For example, if the user clicks on average and max power, then the code blocks needed to get the average and the max power are done.

For the third question, the function "TriScatteredInterp" is used. This function takes scattered data from data points across the map, and interpolates this data in two dimensions. This interpolation is set to the dimensions we set the graph to. We then used the contour function over a picture of a map of UC Davis loaded from the GUI. The flow of the second works like this (next page):



ENG6: Final Project YouTube Link - ENGTeam6
http://www.youtube.com/watch?v=HXjklc8CZ-E