Background

Since its release, Apple’s iPhone has changed the world of cellular phones, providing remarkable connectivity and an application framework that brings embedded devices one step closer to their desktop counterparts. Among its capabilities, the iPhone’s core location framework provides relatively accurate location data using a combination of Wi-Fi technology, cellular networks, and most recently, assisted GPS. Many applications such as Google maps and directions use these services to provide the user with detailed data. One can imagine the types of applications that can be created using the combination of Wi-Fi access and the location data. For example, parents could track the location of their teenager at any point in time, a form of parental control. Also, a woman alone at night could transmit her location to a website in case of an emergency situation such as kidnapping.

While the iPhone has likely exceeded most people’s expectation of the capabilities of mobile devices, the iPhone still has its limitations, specifically with energy. The iPhone runs on battery power and the location services are power hungry, especially GPS. An application running on the device should be aware of the energy capacity of the system and modify its performance accordingly, possibly reducing the accuracy or frequency of location updates.

Goal

For our project, we are going to create a tracking system using the iPhone. The overall goal is to use several different locating services to track the location of the system. We will explore using the built-in iPhone core location services as well as a GPS module. To enhance the embedded software, we will use a C++ program or a website to receive data from the system and display to a user.

Approach

Part 1:

For the first part of this project, we will write an iPhone application using the Developer SDK. The application will periodically obtain the location of the system using the built-in and either 1) store in memory or 2) transmit (via Wi-Fi), depending on the current system state. We plan on modifying the frequency of the updates either using data from the accelerometer or a past data prediction algorithm. To display the location of the system, we will write a C++ program that uses sockets etc. to obtain the information.

Part 2:

For the second part of this project, we will use the GPS module to improve the accuracy of location. If feasible, we plan to balance the tradeoff between accuracy and power consumption using both GPS and the built-in location services. Time permitting, we hope to enhance our application by using a website in place of the program.
Progress

Hardware:
After obtaining the iPhone, we used the Windows version of the pwnage software [1] to jailbreak and unlock the phone. Jailbreaking allows the phone to function without an AT&T contract, while unlocking enables custom applications to be loaded and used. The software creates a custom firmware using the basic iPhone restore firmware and the bootloader binaries. We restored the phone using this custom firmware, so currently, the iPhone has wireless connectivity and can obtain location data using Wi-Fi functionality.

We also ordered and received the iPhone compatible GPS module (shown below) from SparkFun [2]. Because there is little to no documentation for this module with the iPhone, it will require a custom application to be written. Fortunately, there is extensive documentation on the SiRF III EM-408 GPS module [3] itself, so we only anticipate (a lot of) difficulty in creating a software interface to interact with the hardware.

Application Environment:
Using the iPhone program portal, we obtained a certificate and provisioning profile and certified both the borrowed iPhone and a personal (3G) iPhone. We downloaded and explored the iPhone SDK, testing sample applications in XCode and the simulator. We also attempted to load a simple test application onto one of the devices but have not yet succeeded due to verification errors.

Software:
In order to communicate between the desktop and the iPhone, we wrote both an iPhone application and a desktop (C++) program using BSD sockets. The desktop program creates a socket and listens for connections. The iPhone application (an extension of the LocateMe example application [4]) also creates a socket and then sends the location data to the desktop socket at a given IP address each time the data is updated.
The desktop program displays this information to the screen. Example screenshots of this interaction are shown below.

Future Work

Our next step is to download our current application to both of the iPhones. Once this succeeds, we will begin to modify our program to incorporate some of the other features listed in our approach. We plan to use the Google maps API [5] to display the received data in a user friendly format. Finally, we will attempt to use the GPS module to obtain more accurate data on the iPhone 2G.

References