Commuter Tracking Sensor Network

Design of Experiments

"A wireless sensor net for outdoor tracking and localization."

Project Principles:

Seth Hendrick
Alessandro Sarra
Jared Mistretta

Project Collaborators:

Dr. Jeffrey Wagner - mjwgse@rit.edu

Submission Date:

May 5th, 2014

Testing Strategy

Signal Acquisition and Conditioning

Testing of the power acquisition and conditioning circuit will involve five phases as shown in Table 5. Phase 1, early testing, will involve varying the length of the Windbelt to maximize AC current production and minimize length. Subsequent tests of the early stage involve varying fan speed and the angle at which wind is applied.

*(**Warning: All Lithium-Polymer battery testing MUST be accomplished with the battery in plain site prior to remotely testing. Li-Po batteries can become highly unstable if charging conditions are less than nominal.**)*

Test ID	Phase -Component	Description	Pass Condition
P.1	Early - Windbelt Component	Vary length of the belt and measure AC voltage peak-peak using an oscilloscope.	AC current production is at a maximum, size of Windbelt is at a minimum.
P.2	Early - Windbelt Component	Vary fan speed and measure AC voltage using oscilloscope.	AC current production is maximized.
P.3	Early - Windbelt Component	Vary the angle at which the optimal wind speed is applied and measure AC voltage.	AC current production is maximized.
P.4	Intermediate I - AC/DC Rectification Component	Connect optimal design from early testing phase to half-wave rectifier circuit. *(Requires completion of windbelt.)*	Smoothing levels are adequate, and levels are held at 80% or higher of 3.3V with minimum fall time.
P.5	Intermediate I - AC/DC Rectification Component	Connect optimal design from early testing phase to full-wave rectifier circuit. *(Requires completion of windbelt.)*	Smoothing levels are adequate, and levels are held at 85% or higher of 3.3V with minimum fall time.

P.6	Intermediate II - Primary Boost Converter Component	Connect output of rectifier to boost converter at VIN_DC port (pin 2). Measure output of boost controller at VSTOR (pin 15) using an oscilloscope (*Requires completion of custom PCB power management module.*) When running without MPPT, reference Figure 4 in the bq25504 technical document for the proper port configuration	Measure voltage levels at output over elongated periods of time. Levels should not exceed those allowable by the battery (4.2V). Observe and track the DC output signal as it fluctuates over time to determine whether MPPT is necessary. Verify that the boost controller remains on during harvesting periods.
P.7	Intermediate II - Lithium Polymer Battery Component	Manually charge the Li-Po battery at the specified voltage for a nominal charge period. Measure the output voltage with a multimeter. *(May be done in an automated fashion as well.)*	Measure voltage levels of the battery periodically until it is determined that the battery has fully discharged. Record the time it took for the battery to discharge.
P.8	Intermediate II - Voltage Regulation Component	Connect fully charged Li-Po battery to voltage regulator on custom PCB. Test DC output with multimeter at both outputs. *(Should be completed after battery testing)*	Measure voltage levels at both ports to ensure that 3.3V is being provided to the XBee and 3.7V is being provided to the second boost converter (6-10V out).
P.9	Intermediate II - Secondary Boost Converter Component	Connect 3.7V pin to secondary boost converter and measure voltage at the output using a multimeter. Do not configure MPPT yet. *(Not sure which boost converter will be used yet)*	Measure voltage levels at the output of the boost converter. Output should be between 6 and 10 V to adequately supply the Cortex processors. Observe the output over time to determine whether MPPT is required at the secondary boost converter level.
P.10	Advanced I - Primary	Connect the battery to the boost converter as shown in Figure 4 of the bq25504	Battery duration, charge level and output should be within 20% of the

	Boost Converter Component	technical document. Measure duration and level of charge without MPPT enabled across the poles of the battery and at pin 15. *(Should be completed after battery testing. Do not leave the battery unattended!)*	measured discharge period of the battery when measured separately. Compare discharge rate with those of the battery without the sustainable energy source.
P.11	Advanced I - Primary Boost Converter Component	Configure the boost converter as shown in Figure 2 of the bq25504 technical document. Measure duration and level of charge with MPPT enabled for a solar energy application with the Li-Po battery attached. *(Should be completed after battery testing. Do not leave the battery unattended!)*	Battery duration, charge level and output should be within 20% as above for elongated periods. Compare discharge rates with those of the battery without the sustainable energy source, and with the source but without MPPT.
P.12	Advanced I - Secondary Boost Converter Component	Configure the boost converter for use with MPPT and measure the output of the converter using a multimeter. *(Should be completed after non-MPPT testing if required)*	Measure voltage at the output over elongated periods to ensure that the signal provided is within 6-10V, and that adding MPPT further stabilizes the signal for use with the MCU.
P.13	Advanced II - Secondary Boost Converter Component	Configure the boost converter for use with MPPT and connect the M0 and M4 processors. Measure voltage at the output using a multimeter. *(Should be completed after MPPT testing without load.)*	Measure voltage at the output over elongated periods to ensure that the signal provided is within 6-10V.
P.13	Advanced II - Integration with Focus on Signal Acquisition	Deploy a single node in a controlled environment and monitor power conditions closely. *(Should be completed after all other component testing.)*	Measure voltage levels, battery duration and charge throughout the elongated period, and confirm that CV algorithms, networking, and all power dependent subsystems are fully-functional.

P.14	Advanced II - Integration with Focus on Signal Acquisition	Deploy a single node in an outdoor environment. Allow node to operate without intervention and monitor conditions remotely. *(Should be completed after controlled integration testing.)*	Measure voltage levels, battery duration and charge throughout the elongated period, and confirm that CV algorithms, networking, and all power dependent subsystems are fully-functional.
P.15	Advanced III - Acceptance with Focus on Power Management Requirements	Deploy multiple (3-4) nodes in an outdoor environment. Allow node to operate without intervention and monitor conditions remotely. *(Should be completed after controlled and single-node integration testing.)*	Measure voltage levels, battery duration and charge throughout the elongated period, and confirm that CV algorithms, networking, and all power dependent subsystems are fully-functional.

Table 1 - Signal Acquisition Test Descriptors

Intermediate testing focuses on continuing testing with the optimized design chosen from phase 1. The rectifier circuit is connected to the windbelt, and smoothing and fall time are optimized. For the second intermediate phase, the focus is on providing safe levels at the output of the boost converter with the MCU connected to a non-battery power source.

Advanced testing allows for connection of the battery and MCU to the boost converter and deployment in an outdoor environment. Battery voltage levels, duration, and charge levels will be measured for as long as the battery maintains its charge. Once components are proven fully functional, the signal acquisition system will be tested as a whole in a controlled testing environment. Acceptance testing will be performed in the field, most likely with less distance between nodes for easy maintenance and monitoring prior to actual deployment on the Lehigh Valley Trail. Tests will be run to ensure that all power-dependent requirements are satisfied after integration level testing has completed.

Computer Vision Testing

For the software side of the nodes, unit testing can be used for all software written while development is underway. Unit testing prevents bugs from escaping into the field, as it ensures all functions and branches work as intended separate from all other modules.

To test the software algorithm that determines what walks by a node, the nodes can be set up in a controlled environment, such as inside of a lab. A person can then walk by with or without a bike, and the node should be able to identify if the person is walking or riding a bike. The subject should walk past the node at various speeds, and the node should still be able to make a capture. The node should be able to pass captured information to the rest of the nodes, and to the gateway so it can be recorded by the database.

Various lighting scenarios should be tested as well. In a lab environment, the lighting should be adjusted so it gets dimmer until the node is unable to make accurate readings. This will be the minimum light needed for operation, and the node might not require frames to be captured at times like this in the field. During the lab test, a light should be positioned at various angles, such as behind the node shining on the target, behind the target shining on the node, overhead, and to the left and right of the node. Regardless of where the light is positioned, the node should still operate accurately.

When the node is built, and the software written, a stress test that can be performed is to deploy the nodes on the RIT quarter mile, which gets a lot of foot traffic. This test will also show just how accurate and fast a node is. If the node just can not keep up with the traffic, some redesigning might need to occur.

Test ID	**Phase**	**Description**	**Pass Condition**
V.1	Early - Image Sensor Component	Verify that the image sensor is correctly capturing images by connecting the Pixy Cam	Images in front of the lens are rendered correctly in the Pixy Cam GUI window.

		via USB, loading the GUI and observing.	It can be assumed that the M0 is functioning correctly.
V.2	Early - Image Sensor Component	Verify that the image sensor is correctly identifying colors by running the included color identification algorithms with the board connected via USB	The chosen color is amplified in processed images, as shown in the Pixy Cam GUI window. It can be assumed that the M0 is functioning correctly.
V.3	Intermediate - Infrared Component	Connect the board via USB and connect an Infrared sensor to the camera via GPIO or UART. Dim the room of all light sources. Verify that the sensor data is correctly displayed in the Pixy Cam GUI window by walking in front of the camera and observing..	The image should be correctly represented with more orange to red colors showing higher temperatures. The background should contain blue and purple colors for lower temperatures.
V.4	Intermediate - CV Algorithm Component	Compile and load the CV software onto the Pixy Cam. Walk by the Pixy Cam to verify whether the algorithm is functioning properly. *(To be tested with and without Infrared sensor connected.)*	The image should be identified and the correct CV size capture, speed and direction determinations are made.
V.5	Intermediate - CV Algorithm Component	Vary speeds at which the subject walks by the Pixy Cam to verify that the algorithm still functions properly. *(To be tested with and without Infrared sensor connected.)*	The image should be identified and the correct CV size capture, speed and direction determinations are made.
V.6	Advanced - CV Algorithm Component	Walk by the Pixy Cam with a bike or other large object to verify that the algorithm still functions properly. *(To be tested with and without Infrared sensor connected.)*	The CV algorithm is able to correctly discern between a walking and biking commuter based on speed of the individual, and the correct output displays in the GUI.

V.7	Advanced - CV Algorithm and Infrared Component Integration	Compile and load the CV software onto the Pixy Cam. Dim the lights in the room. Walk by the Pixy Cam to verify whether the CV algorithm is functioning properly. *(To be tested after CV and IR component tests pass.)*	The algorithm should be able to use the Infrared data with the lights dimmed to accurately detect size, direction and speed of an individual.
V.8	Advanced - Computer Vision and Networking System Level Integration	Compile and load the CV software onto the Pixy Cam. Connect the cam via USB. Connect the CMUCam to the XBee as well via GPIO or UART header and provide power to it via pre-charged Li-Po battery connected to custom voltage regulation unit. Dim the lights in the room. Walk by the Pixy Cam to verify whether the CV algorithm is functioning properly. Turn on the lights and perform the same tests. *(To be tested after CV, IR, voltage regulation and GPIO component tests pass. The website GUI should also be operational, but data can alternatively be viewed through local GUI.)*	The algorithm should be able to use the Infrared data with the lights dimmed to accurately detect size, direction and speed of an individual. The algorithm should also work with lights on. This data should be broadcasted to a central node, server, and pushed to a website to view for confirmation.
V.9	Advanced - Computer Vision and Power Management System Level Integration	Compile and load the CV software onto the Pixy Cam. Connect the CMUCam to the secondary boost converter. *(Warning: Do not perform this step until component level testing of the secondary boost converter is complete.)* Connect the CMUCam to the XBee as well via GPIO or UART header and provide power to it via pre-charged	The algorithm should be able to use the Infrared data with the lights dimmed to accurately detect size, direction and speed of an individual. The algorithm should also work with lights on. This data should be broadcasted to a central node, server, and pushed to a website to view for confirmation.

		Li-Po battery connected to custom voltage regulation unit. Leave out everything prior to the battery in the Signal Conditioning circuit. Walk by the Pixy Cam to verify whether the CV algorithm is functioning properly. Turn on the lights and perform the same tests.	
V.10	Advanced - Computer Vision Acceptance Testing	Compile and load the CV software onto the Pixy Cam. Connect the CMUCam to the secondary boost converter. *(Warning: Do not perform this step until component level testing of the secondary boost converter is complete.)* Connect the CMUCam to the XBee as well via GPIO or UART header and provide power to it via pre-charged Li-Po battery connected to custom voltage regulation unit and add signal acquisition and conditioning circuit including primary boost converter, rectifier, and windbelt. Walk by the Pixy Cam to verify whether the CV algorithm is functioning properly. Turn on the lights and perform the same tests. Vary size of CV subjects. *(Perform this test after System level integration is complete for each system involved.)*	The algorithm should be able to use the Infrared data with the lights dimmed to accurately detect size, direction and speed of an individual. The algorithm should also work with lights on. This data should be broadcasted to a central node, server, and pushed to a website to view for confirmation. Extended testing should be performed at this level once all system level and integration is complete. The node should be able to process CV algorithms for an extended period with sustainable energy available.

Table 2 - Computer Vision Test Descriptors

Encasement testing

The node will be deployed outside in Rochester's infamously bad weather. Therefore, a test to ensure the node remains intact in extreme weather needs to occur once the weather-proof chamber is completed.

Test #	**Test Name**	**Description**	**Reason**	**Pass Condition**
E.1	Temperature Test	The nodes should be placed in a cold environment to simulate Rochester's winter, and a hot environment to simulate Rochester's summer.	Rochester has a variety of weather conditions, that can range to several degrees below zero, and up to almost 100 degrees. The nodes will be outside during these times, and should be able to survive in them.	If all the node's hardware still works after being in the extreme environments, the test passes.
E.2	Rain Test	Water can be sprayed on the encasement to simulate rain, and a powerful fan can be used to simulate consistent wind. For these tests, paper towels or some kind of other material that reacts to water should be placed inside the encasement instead of the expensive hardware.	Rochester can have days of rain in a row. The housing needs to be able to keep the water out at all costs, otherwise the hardware might get fried.	If no water gets into the encasement, the test passes.

		This way, if water does manage to get in, no hardware is damaged.		
E.3	Drop Test	There should be a drop test of 5-6 feet to ensure that the trail node's encasement is sturdy enough to protect the hardware inside. This can be done by placing a raw egg, or something else cheap and easily breakable, in the encasement instead of the expensive hardware.	It is possible that while deploying the nodes, they could be dropped. The encasement should protect the hardware from drops.	If the egg cracks when being dropped, then the encasement is not sturdy enough, but if the egg is intact, the the test passes.

Table 3 - Encasement Test Descriptors

Network Architecture Testing

It is essential that all the nodes are able to communicate with each other over a long distance. To ensure that all the nodes will be communicating with each other through the XBees, the following tests can be done:

1. The first test that should be performed is a small scale test to ensure that the XBees are able to be configured through UART and send messages via the DigiMesh protocol. This test should be performed once the XBees are acquired.
    a. Acquire 4 XBees.
    b. Attach the 4 XBees to either Raspberry Pis or the Pixy Cam through UART. A mixture of Pis and Pixy Cams is preferred, as the Pixy Cam will be the MCU for the trail nodes, and the Pi will eventually act as the gateway node.
    c. Ensure the XBees can be configured through UART. If they are unable to be configured, then the XBee Development kit might need to be purchased in order to program the XBees.

    d. Ensure the XBees can send and receive messages from each other. If they are able to, then the XBee Development Kit is not needed.
    e. For this test, the XBees can be in close proximity with each other. Test 3 increases the distance.
2. The second test that should be performed is a test to see how far away the XBees are able to communicate with the selected antenna. This should be done once the previous test passes.
    a. Create 2 nodes by connecting an XBee to either a Pi or a Pixy Cam through UART.
    b. Make one node stationary, and take the other node and go as far away from the stationary node as possible before communication cuts out.
    c. The biggest distance between two nodes on the trail is about 1.75 miles, so the XBees should be able to communicate with each other from at least 2 miles apart.
    d. If this test fails, then a more powerful antenna needs to be purchased.
    e. Preferably, the test should be performed on the trail itself, as that's where the nodes will eventually be placed.
3. The final test is a small scale trail test. This is where a subset of the trail has nodes deployed to it. If this test passes, then it is probably safe to purchase more nodes and cover the entire trail. This test can be performed when the last test passes.
    a. Attach 4 XBees to either Pis or Pixy Cams.
    b. Deploy 4 nodes are positions 17, 2, 3, and 4 (referring to Figure 2). These four positions are the most spread apart spots on the trail.
    c. If all the nodes can communicate with each other, then the test passes, and it is probably safe to up the scale to cover the whole trail.
    d. If the test fails, then some of the nodes might need a more powerful antenna.

Server Security Testing

Once the Raspberry-Pi server and all security implementation is in place, an attempt will be made to take control of the server from an outside source using DDOS, SQL injection and brute forcing. If this attack succeeds, depending on the amount of time and difficulty it took to penetrate server security, additional security measures may be taken.

Test #	Test Name	Description	Reason	Pass Condition
S.1	Ping test	The router should not be pingable from an outside source.	When many people try to ping a server, it could create a DDOS attack. Therefore, pinging should be removed to prevent this from happening.	Trying to ping the router from outside the network should not work.
S.2	Disable Root Login	While SSHing into the server, users should not be able to login as root	Root is all knowing in linux. If someone were to somehow gain access to root, they could wreak havoc on the server.	Trying to login as root results in an access denied.

S.3	SSH non-standard port test	While trying to SSH into the server using the default ssh port, the result should be not being able to connect.	By moving the SSH port to a non-standard port, it makes it difficult for some hackers to find the ssh port, and try brute forcing.	Trying to login to port 22 via ssh will fail. Trying to login to the non-standard port will succeed.
S.4	Disabled Password Test	Trying to login with SSH will have an access denied without an ssh key. SSHing will not be allowed without a password.	SSHing with a user name and a password is not safe for a server. A hacker can brute force their way in.	SSHing into a server without a key will result in an access denied. SSHing will not ask for a password.
S.5	White hat hacker test	Borrow one of the many security majors the group knows, give them the ip address, and have them attempt to take the server down.	Having a friendly hacker attempt to break into the server will emulate a more sinister hacker trying to get in. If any information is compromised by the friendly hacker, they will not steal it.	The hackers should not be able to take the server down.

Table 4 - Server Security Test Descriptors

Website Testing

The website is the gateway to the data. However, it can also be a gateway for a hacker wreck havoc. These tests will ensure that anyone with dark intentions will not be able to compromise the server from the website.

Test #	Test Name	Description	Reason	Pass Condition
W.1	Sanitation Test	Data from the website is sent to the server via http get and post requests. Some commands sent that could break the server include “; && rm -rf /” and “drop table *;”. These commands should of break the server.	If the data is not “sanitized” correctly, it is possible for a user to drop a table from the sql database, or execute shell commands.	Posting and getting from the server using common commands that break a server should .,not break the server.
W.2	Password protection test	Connect wireshark, and see if user names / passwords are able to be seen leaving the client and going to the server.	If the website is meant to have a limited set of users, then a user name and a password is needed. The passwords need to be protected.	The password leaving the client should be hashed and salted so a hacker can not steal them.
W.3	User login test	Ensure valid users can log in and invalid users can not log in	The data needs to be available to a limited set of users.	Valid users can log in, invalid users are rejected.

			Therefore, a user name and a password is needed to gain access to the website.	
W.4	White-hat hacker test	Borrow one of the many security majors the group knows, tell them the website, and see if they can break in.	Having a friendly hacker attempt to break into the website will emulate a more sinister hacker trying to get in. If any information is compromised by the friendly hacker, they will not steal it.	If the friendly-hacker can not break in, the test passes.

Table 5 - Website Test Descriptors