# Remote Measurement and Monitoring Critical Design Review

## Johns Hopkins University EN 525.743 / New Spin

Wesley Myers

**ABSTRACT**

This document lays out the requirements for research being conducted for having a laser range finder system manually and automatically measure distances to targets, as well as monitor them within the system's field-of-view.

## Table of Contents

## Project Goal and Description

The goal of this project is to enable users to remotely measure noticeable change using a laser range finder. This fulfills requirements from New Spin[1] customers who want the ability to remotely measure distance to track changes at electric substations that cannot otherwise be accomplished using image differentiation. In addition, the system could be used for absolute measurement for data to go into New Spin's You Control[2] system.

**Figure 1 - Example Electric Substation**

As shown in Figure 1, electric substations are often remote and thereby vulnerable to nature or thieves. Thus having an automated system that can measure distances to targets provides the customer with an additional measurement and the confidence that their assets are safe.

This project is being done in conjunction with a class[3] at the Johns Hopkins University. As such, the requirements are being set to meet a semester timeline.

### High Level Requirements

The system must have a web user interface to control it. The user must have the option to manually select targets for automatic measurement, or ask the system to determine what targets might be of interest for the user. Once the targets are selected, the system should then be able to periodically measure those points. Security system applications may also be applied to this system if time permits. This

[1] www.newspin.com
[2] http://vimeopro.com/newspin360/youcontrol
[3] http://apps.ep.jhu.edu/course-homepages/3088-525.743-embedded-systems-development-laboratory-houser

system can be hard wired via Ethernet, or be deployed wirelessly for connection and control.

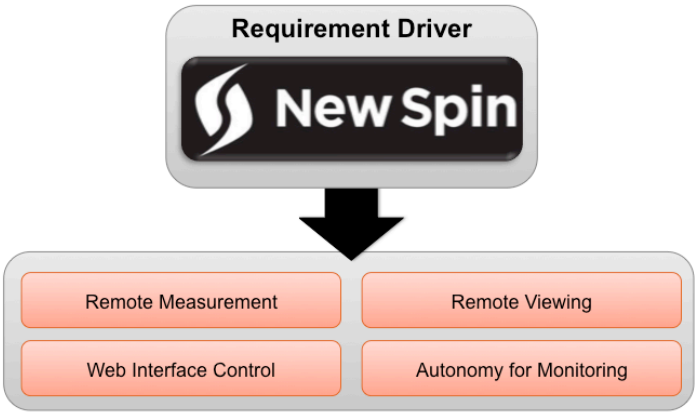


Figure 2 - High Level Requirements

In order to fulfill this requirement, an embedded system platform must be created to allow remote viewing and sensing. As a prototype, the Raspberry Pi will be used for rapid development. This will support a camera, servos, and a range finder, which are all necessary for this project.

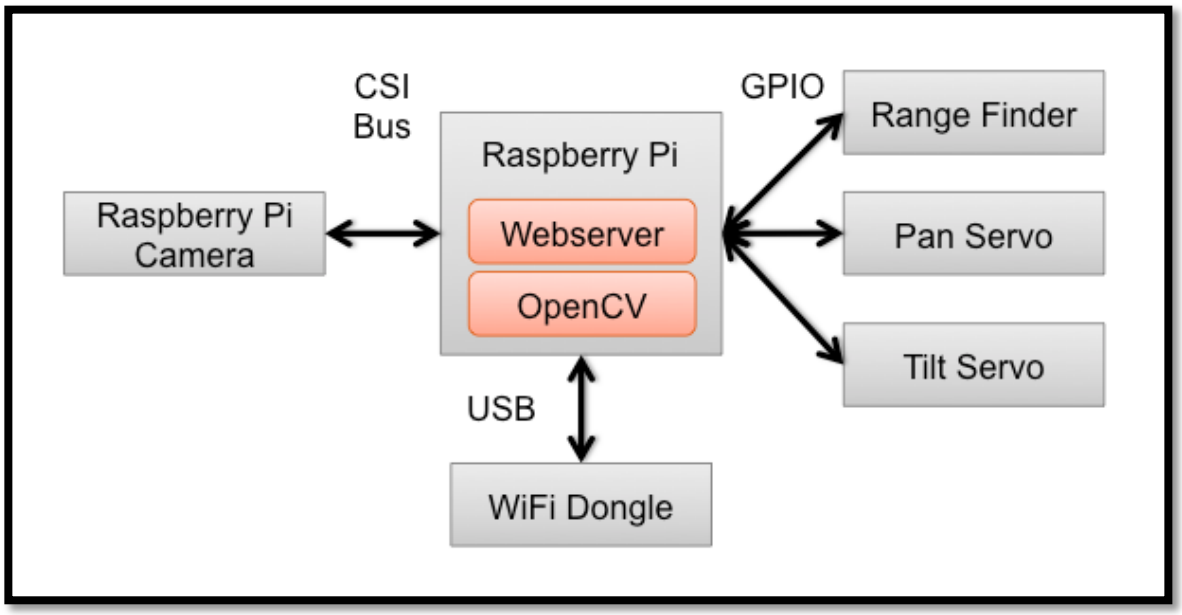


Figure 3 - System Architecture

# System Components

## Raspberry Pi

This is a credit-card sized computer with peripherals that users are used to on a typical computer. This board allows for development with more horsepower than an Arduino. In this case, image processing will be done utilizing OpenCV to automatically locate targets of interest. Further discussion into the OpenCV algorithms being used can be found in later sections.


Figure 4 - Raspberry Pi Model B+

## Raspberry Pi Camera Module

This camera is specifically designed to interface with the Raspberry Pi's CSI bus. There are two cameras to choose from: standard RGB and Infrared. The IR platform becomes of interest to verify the location of the laser, as well as allow the user to operate the system at night.

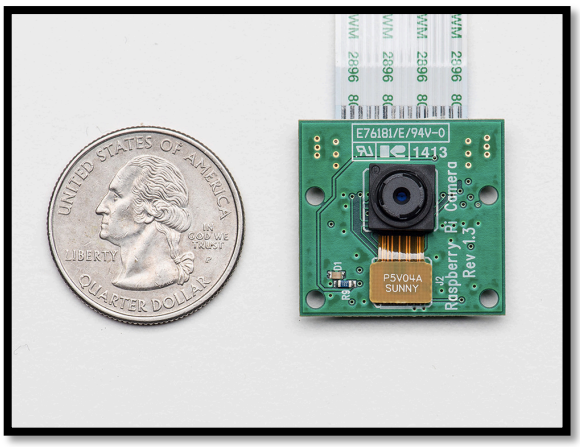


Figure 5 - Camera Next to Quarter

## Laser Range Finder

The SF01 Laser Range Finder[4] is a platform designed by LightWare Optoelectronics. It is a relatively low cost laser range finder that can measure targets at up to 60 meters. It also comes in a small package that is ideal for the system. For commercial grade range finders, this is the only one within the required specifications that is able to interface with a Raspberry Pi or Arduino platform.

Figure 6 - SF01/A LRF

## Pan-Tilt System

To save time on design of this system, we'll target a pre-fabricated pan-tilt system that is tried and true. Servo City makes an assortment of pan-tilt platforms to choose from. The SPT200[5] model in particular can hold up to 2-pounds, which should be adequate for the system's purposes. To operate this system, two HS-5685MH[6] servos will be used as recommended in the SPT200 description.

Figure 7 - SPT200 Pan-Tilt Platform

[4] http://www.lightware.co.za/shop/en/lrf-modules/31-sf01a.html
[5] https://www.servocity.com/html/spt200_pan__tilt_system.html
[6] https://www.servocity.com/html/hs-5685mh_servo.html

## Web Interface

The UI must be able to let users operate the system. Many of these options turn into buttons and separate pages that the user can manipulate to operate the system. The following tasks must be accomplished via the UI.

- Tilting the system
- Panning the system
- Viewing the video feed
- Active measurement
- Autonomous target selection options
- Display measurement results

The webserver that will be used is Django[7], which is an open source web application framework. Many websites, such as Mozilla[8] and Instagram[9], use this framework. It encourages rapid development while adhering to stringent web requirements. This is all written in Python, which is the Raspberry Pi's preferred language.

There are already libraries[10] out there that allow easy import of Django's libraries onto the Raspberry Pi.

### UI Examples

The home page will be where the user interacts with the system. Here the user can pan-tilt the system, view the camera feed, measure the distance to a target, as well as create a Target List. The Target List will be essentially the list of points that the system will actively measure.

[7] https://www.djangoproject.com/
[8] https://www.mozilla.org/en-US/
[9] http://instagram.com/
[10] https://github.com/phalt/DjangoPi

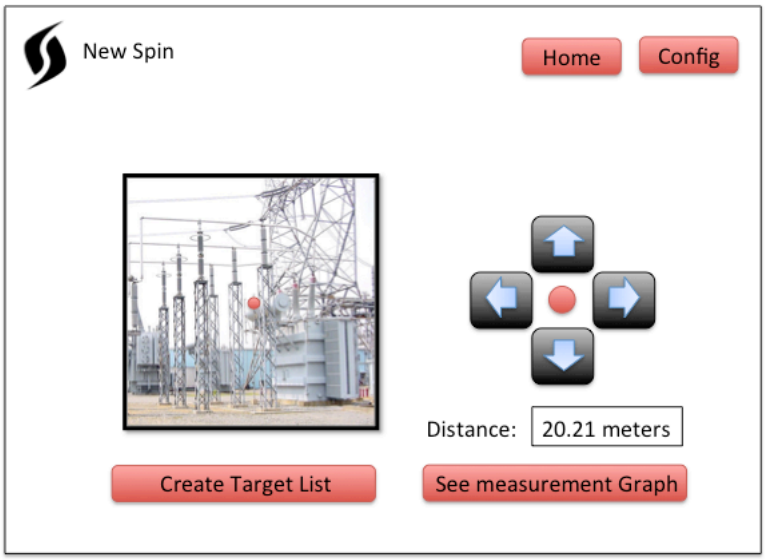


Figure 8 - Home Page

The Configuration Page allows the user to set up automated detection, which is where the system will check over a list of targets based on a Target List. The user can also set up automatic notifications such that the system does not need to be observed at all times.

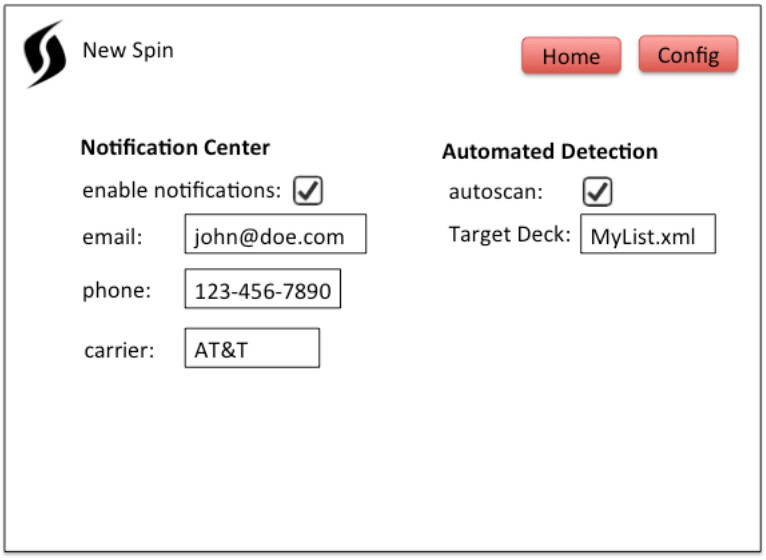


Figure 9 - Configuration Page

## Functional Operation

As described in the requirements section, there must be three main modes to this system. The first is to have the user select a target to measure within the field of view. The user can click on the image to have the pan-tilt move, or manually do it themself. At this point, the measurement algorithm is started and then the resulting information is displayed to the user.

The other two modes are automatic measuring modes. These modes measure a series of points from which the user can get automatic updates if there is a change.

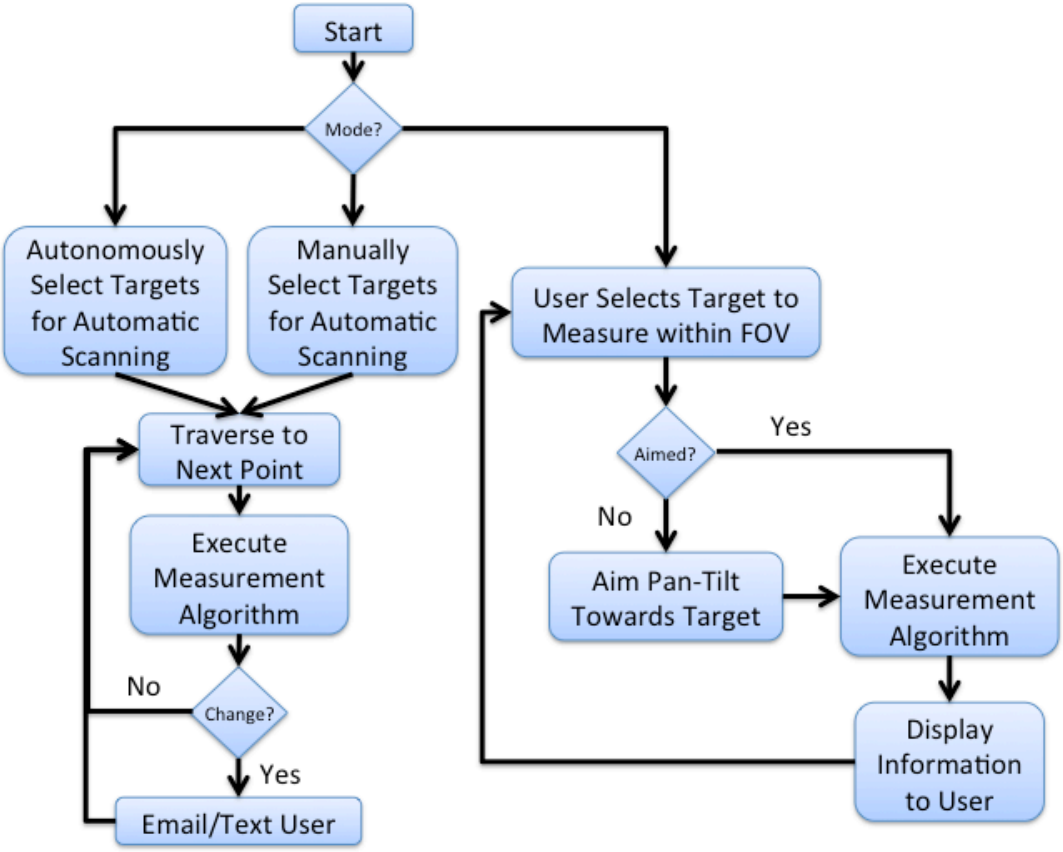


Figure 10 - High Level FSM

In the Autonomous Automatic Scanning Mode, the OpenCV algorithms are used to determine points of interest for the user. The user can then select from this list of suggestions. Next, the pan-tilt will aim the system at these targets, measure the distance to create a profile of the target, and then the information is saved for later comparison to the Target List. Once the series of desired points are exhausted, the system will go into an automatic scanning state. Here the system will execute the measurement algorithm over these series of points. If a change is detected, then the user will be notified.

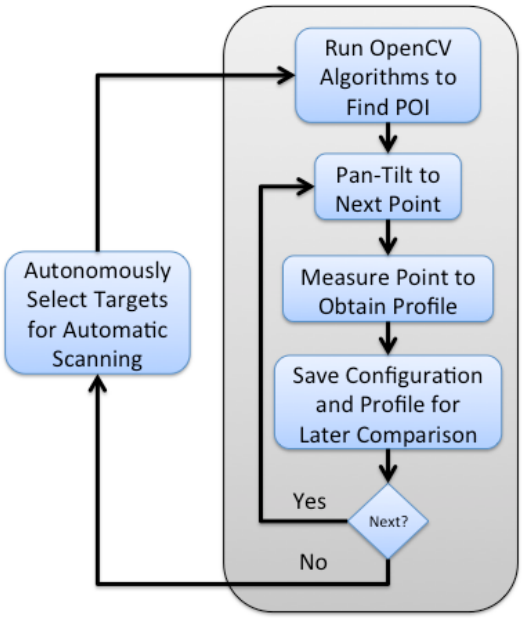


Figure 11 - Autonomous Point Selection

In Manual Automatic Scanning Mode, the user will be prompted to determine which targets are of interest to them. As the user measures the distance to create a profile of the target, the information is saved for later comparison. Once all points are selected, the system will go into an automatic scanning state, identical to the past described mode. Here the system will execute the measurement algorithm over these series of points. If a change is detected, then the user will be notified.

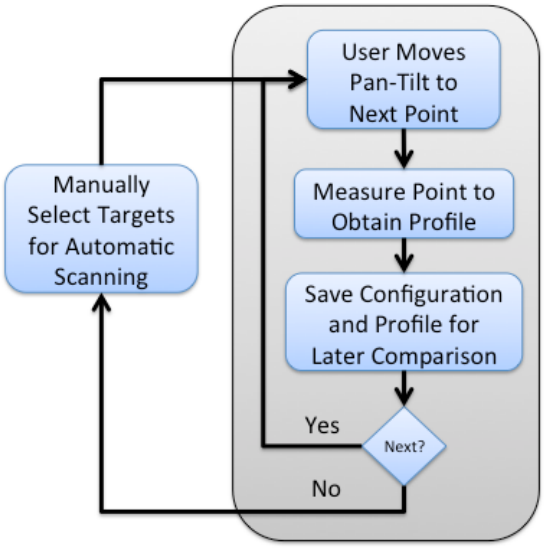


Figure 12 - Manual Point Selection

## Risks and Challenges

### Measurement of Targets

The first step of this project uses servos, which are limited to 180 degrees of motion on 1 degree fixed steps. This may prove to be quickly impractical in implementation due to the limited step size, and thus stepper motors might be more desirable for more precise movement. The further the distance for measurement, the more precision in movement is needed for proper measurement. So in this classroom exercise, servos will be sufficient. As the distance of the laser increases, the more "steps" will be required if the system uses stepper motors. This problem will persist as the system's laser is upgraded.

The use case example here is where a user wants to measure the distance to the power line. In this case, the power line is 2 centimeters thick and is 50 meters away. The following diagram illustrates this use case.

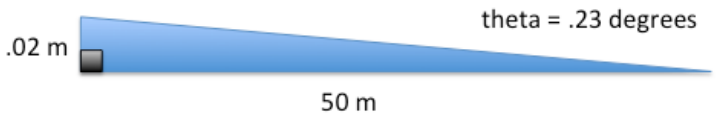


Figure 13 - Case Study Example

Given that we have a .23-degree window to hit our target, we need a stepper motor that has about 3600 steps unless we want gearing to be applied to the system. That will be the only way to accurately point the laser, assuming it is lined up properly.

Figure 14 - Example of a Power Line

Ways to mitigate these issues for a servo system, or high step stepper motor, is for the system to scan ithe segment of motion. So in this use case, the laser would be passing over potentially free space when it encounters the wire. Thus a blip in the scan line will occur, which is the wire. This will further be discussed in the algorithms section.

### Integration of Laser

Though LightWare might advertise the SF01 as a sensor that is interface-able with Raspberry Pis and Adruinos, there is no documentation of anyone attempting to do so with a Raspberry Pi. The interface protocols for the LRF are very well documented, which should allow initial guidance with the laser integration. LightWare does not provide any source code for interfacing with a Raspberry Pi, however there is code[11] for an Arduino.

Ways to mitigate this would be to have the Raspberry Pi interface with an Arduino, which will interface with the LRF. Integration between a Raspberry Pi and Arduino is well documented and can be accomplished.

### Servo Operation

The Raspberry Pi is not designed with any Pulse Width Modulation pins, or Analog/Digital pins that the Arduino comes with. This means bit-banging must be done with the Raspberry Pi to simulate PWM signals to drive the servos.

An Arduino could be used to drive the Servos. As mentioned in the previous sub-section, there is plenty of documentation on how the Raspberry Pi can interface with an Arduino.

### Identifying the Laser

One of the ways to verify the system is functioning properly to identify the laser in the image. At close ranges, this should not be a difficult task. However, at longer ranges, the laser might not be easily seen.

Ways to mitigate this is to place a magnifying lens on the camera, however that would reduce the field of view for the camera. At short distances, we can verify the laser position since the laser should be close enough to be visible. This ensures that the laser and camera are co-aligned.

### Webserver

This will be the first time that the lead engineer on the project will be designing a web server. Though there is a lot of documentation on Django, initial understanding will impede progress.

[11] http://www.lightware.co.za/shop/en/content/8-software

# Algorithms

## Object Measurement

As the goal is to measure the distance to various targets, we must come up with an algorithm to verify this. The simplest method would be to simply measure a target using multiple samples and then run the information through a low pass filter. In this case, we cannot simply do this. As the step-size of the pan-tilt servos limits us, there is a high probability that we will not be aligned with the desired target. Thus we must be creative in how these measurements are taken.

In Figure 14, we have an example use case where the customer may want to measure the distance to a power line. In this case, it could be for verification that the line is still up or for making sure there is no sagging in the line. As explained in the Risks and Challenges section, it is possible that we could miss this target if the system is off by a little bit.

In Figure 15, we have a potentially scan-able area for the laser. In this case, we have four corners labeled as Top Left (TL), Top Right (TR), Bottom Left (BL), and Bottom Right (BR). As the laser moves 1 degree from corner-to-corner, it could do multiple scans per movement. So for example, between TL and TR, it could take multiple samples to cover an object in the middle of the scan.

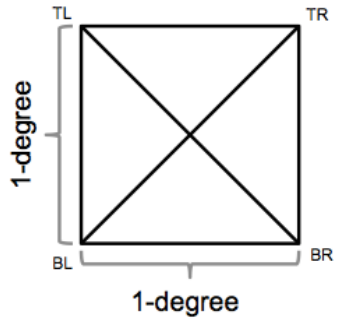


Figure 15 - Range of Measurement

Figure 16's gray line is an example of the power line from Figure 14. In this case, the line is a bit smaller. The red lines represent the laser's movement across the wire.

Figure 16 - Simulated Measurement of Power Line

Figure 17 illustrates the results of the scan lines. In this case, we can see where the power line is detected as shown by the grey blocks. Note that in this case, we're assuming continuous sampling while in implementation, we will have a few discrete samples to evaluate from.

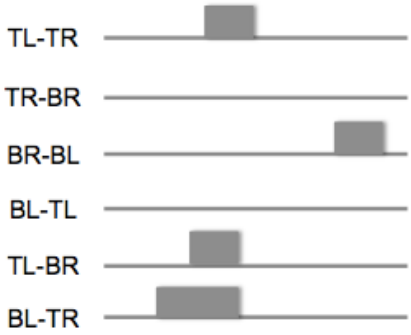


**Figure 17 - Results of Power Line Measurement**

With this information, we can develop a profile of a specified target to remember what it looks like. Alternatively on a scan, we can figure out how far a target is away and provide the user with a probabilistic answer.

## OpenCV

When the user first deploys the system, they can configure the system themselves or allow the system to select targets of interest. In order for the automation to occur, we must utilize various OpenCV libraries to find these targets for the user.

### Scale-Invariant Feature Transform (SIFT)

SIFT[12] is an algorithm that can be used to detect and describe local features in an image. In order to keep the explanation light, this library allows us to make a call to find significant features within the image.

The Scale-Invariant aspect of this algorithm means that the results are invariant to object translation, scaling, and rotation. Meaning that we should have a similar set up results when we apply this algorithm to various substations. This algorithm should reference a database of images, allowing it to select better features on the specified image.

Figure 18 illustrates a possible perspective for the system. In this case, there are many possible features to choose from that might be of interest to the customer.

[12] http://en.wikipedia.org/wiki/Scale-invariant_feature_transform

Figure 18 - Power Substation for SIFT Processing

In Figure 19, the image is processed through the SIFT algorithm. What we get back are a lot of features. In this figure, the features are not filtered. What we see are all of the possible features to choose from. As the algorithm has no perception of free space, one will notice that many centroids of the power structure are actually selected as features. Using this algorithm, we can filter out the insignificant features and offer to the user only the significant features that might be of interest.

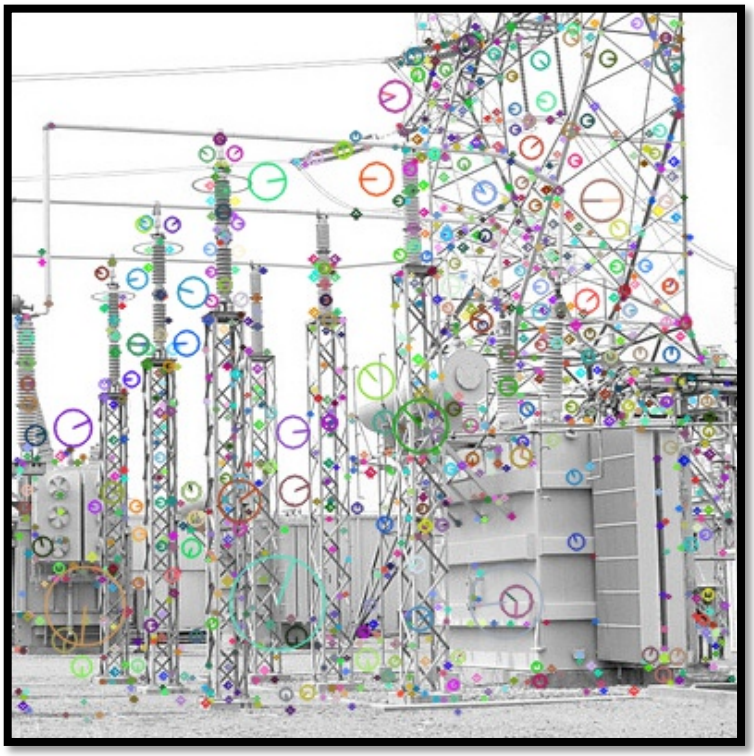

Figure 19 – Key Points Highlighted Pre-Filtering

### Machine Learning Algorithms

A really interesting addition to this system is the ability to identify objects through object recognition. This is no trivial feat for any system, especially on a small platform like the Raspberry Pi. However, libraries exist to be able to conduct basic machine learning algorithms to identify objects.

One such algorithm that may be of interest in this instance is the K-Nearest Neighbor Algorithm[13]. This is a basic classification method that can be done with other OpenCV algorithms that utilize feature detection, such as SIFT. This requires a classification dataset and will require photos of the site, or similar sites, to allow proper classification of the objects. Utilizing the data found from SIFT, we can start classifying objects and build up a database to allow the system to recognize what various objects at the site are.

[13] http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

## Project Schedule

To meet class deadlines, the following timeline has been generated.

| Date | Milestone |
|---|---|
| September 29th | Ordering and Receiving Equipment |
| October 6th | Pan Tilt Mechanism / CDR Due / TDY Tuesday – Friday |
| October 13th | Camera and Laser Integration |
| October 20th | Continue Integration / TDY Sunday - Wednesday |
| October 27th | OpenCV Development |
| November 3rd | UI and Webserver Design |
| November 10th | Algorithm Development / TDY Thursday - Sunday |
| November 17th | Algorithm Development / Vacation |
| November 24th | Testing and Analysis |
| December 1st | Testing and Analysis |
| December 8th | Project Documentation Due |
| December 13th | Final Presentation |

### Order/Receive Equipment

This goal is simply to get equipment for the project. This time also encompasses setting up the development environment for the project, as well as any items that might require further research.

### Pan/Tilt Mechanism

This is one of the simpler aspects to the project. This will just be having a pan-tilt servo system that is integrated with the Raspberry Pi. To be more specific, this involves getting the Pi to bit-bang out a PWM signal to tell the servo where to move. This is because the GPIO for the Pi does not have any specific PWM ports like an Arduino. As there is a work trip that week, it seems like an ideal time to complete this requirement.

### Camera and Laser Integration

This is geared towards creating a platform that aligns the laser and camera onto the system to allow proper alignment. This is given two weeks as to allow a proper test layout to be formalized for calibration.

The platform will need to be robust enough to have the LRF and camera be aligned, as well as support the Raspberry Pi. So machining might be required to accomplish this in a sustainable manner.

At this point, calibration will be done using the IR camera aligned with the LRF. When the LRF is operational, an image will be taken by the system and it will be verified that the laser is lined up with the camera in the center of the frame. The set up will require the laser to be pointed at a wall about 5 feet away.

### OpenCV Development

This block of time is dedicated towards integrating the various OpenCV libraries as described earlier in this document under Algorithms. This section is dependent on the autonomous detection of various features. In addition, blob detection will be required for the calibration of the system.

### UI and Webserver Design

This involves standing up the webserver and creating all of the functionality to properly operate the system.

### Algorithm Development

This is the most difficult part of the project. Using the limited capabilities of the pan-tilt mechanism, we must be creative in identifying targets. This is further described in the previous section on Algorithms.

## Equipment List

The following are core items required to complete this project. There is no immediate price ceiling for this system.

| Item | Quantity | Cost per |
|---|---|---|
| **LRF** | 1 | $500.00 |
| **Raspberry Pi** | 1 | $40.00 |
| **Raspberry Pi Camera** | 1 | $30.00 |
| **Pan-Tilt System** | 1 | $60.00 |
| **Servos** | 2 | $40.00 |
| **Wireless Dongle** | 1 | $12.00 |

### Materials Delivery

Most of the material can be ordered off of Amazon[14]. This includes the Raspberry Pi, camera module, and wireless dongle. These will be delivered by October 3rd as they are all contained within the Amazon Warehouse.

The Laser Range Finder is located in South Africa, but will be acquired by October 3rd. This is by far the most expensive component of this project. In addition, if it breaks, then replacing it will take multiple weeks. The company has these lasers on back-order because they are in such high demand.

The Pan-Tilt system and servos can be ordered from Servo City[15]. These items ship from Kansas City and will arrive by October 3rd.

[14] http://www.amazon.com/
[15] http://servocity.com/

## Reach Goals

Many of the requirements described in this document seem to revolve around security camera applications. A good security camera to model after is the Dropcam[16]. This system in particular has many good features that could be applied here. In this case we can have the system email/text the owner if there is a change. Attached would be an image showing the detected incident. In addition, if videoing capabilities are incorporated, then automatic downloading of recordings of incidents can be hooked up to a Dropbox[17] account. In this instance, Dropcam wants the user to pay for storage, while this system would upload to a free account.

### Notification Saying Recording Has Started

Send an email or text if a target appears to be missing. An image should be included as an attachment so that the user can verify if in fact the target is missing. The user should then be able to provide feedback

### Dropbox Interface

As scans are periodically done, upload images to a specified Dropbox account. This enables the user to check past history, as well as verify that the system is detecting changes appropriately.

[16] https://www.dropcam.com/
[17] https://www.dropbox.com/

## Project Overhead

### Software Management

Any software development will be managed off of New Spin's github[18] account.

### Place of Performance

All development and research for this project will be conducted in the Baltimore, Maryland area.

### Documentation

The JHU class requires that weekly status updates be posted to Blackboard[19]. These status updates will also be posted to Podio[20] as completed. In addition, a final class report will contain detailed instructions on how to design and the build the system. All work will be tracked off of the NSLaser repository in Podio.

[18] http://github.com/newspin
[19] http://blackboard.jhu.edu
[20] https://podio.com/newspincom/ns-dev