Division of Research and Innovation

Technical Documentation Page

1. Report No. CA14-2206	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle: **Crash Attenuator Data Collection and Life Cycle Tool Development**		5. Report Date June 14th, 2014 6. Performing Organization Code AHMCT
7. Author(s): Bahram Ravani, George Burkett, Bradley Leong, and Duane Bennett		8. Performing Organization Report No. UCD-ARR-14-06-14-05
9. Performing Organization Name and Address AHMCT Research Center UCD Dept. of Mechanical & Aerospace Engineering Davis, California 95616-5294		10. Work Unit No. (TRAIS) 11. Contract or Grant 65A0416, Task ID 2206
12. Sponsoring Agency Name and Address California Department of Transportation PO Box 942873, MS #83 Sacramento, CA 94273-0001		13. Type of Report and Period Covered Final Report July 2011 – June 14, 2014, 14. Sponsoring Agency Code Caltrans
15. Supplementary Notes		

16. Abstract.

This research study was aimed at data collection and development of a decision support tool for life cycle cost assessment of crash attenuators. Assessing attenuator life cycle costs based on in-place expected costs and not just the initial cost enhances the selection and procurement process resulting in cost savings for Caltrans Maintenance. The research question addressed included: can a decision support tool be developed based on detailed data that would allow proper estimation of in-service or life cycle rather than the initial cost of crash attenuators? In addressing this research question, first data needs to be collected on actual impact frequency at each crash attenuator site and actual repair costs of each type of attenuators. Data on generic repair costs of different classes of crash attenuators have been gathered based on IMMS (Integrated Maintenance Management System) data and has been used in the form of default values in a decision support system developed as part of this research. These default values can be over ridden by the user based on user input of such estimates. In order to help the user in terms of estimating the impact frequency at a site, a post processor for the IMMS data base has been developed that can search the IMMS data base for a site and obtain the repair frequency as well as actual data on total repair costs. The repair frequency is, however, a lower bound on impact frequency because the crash attenuator at a site may have had other impacts that did not require repairs. This is specially the case for sites with crash attenuators that are less likely to be damaged from impacts. In order to get a handle on the number of impacts that would not require repairs (these are referred to here as nuisance hits), an integrated impact sensor and site monitoring system was developed as part of this research to collect data on such impacts. This system was installed in three different sites and data was collected on nuisance hits. Data from IMMS data base as well as from the three sites with sensors have been incorporated into a decision support system with the acronym "CAL-COST" (Crash Attenuator Life cycle Cost) developed in this study. Input for CAL-COST requires estimates of impact frequency, repair costs and access costs for a site. As output it provides data including the cost break-even point for different classes of crash attenuators that can be matched to crash attenuators available in the market. The CAL-COST decision support tool can also be used to evaluate different Crash Attenuators products at a specific site and their life cycle costs.

17. Key Words: Crash Attenuators; Life Cycle Analysis; Impact Sensing, Decision Support System	18. Distribution Statement No restrictions. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.		
20. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 141	22. Price

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized(PF V2.1, 6/30/92)

DISCLAIMER/DISCLOSURE STATEMENT

The research reported herein was performed as part of the Advanced Highway Maintenance and Construction Technology (AHMCT) Research Center, within the Department of Mechanical and Aerospace Engineering at the University of California – Davis, and the Division of Research, Innovation and System Information at the California Department of Transportation. It is evolutionary and voluntary. It is a cooperative venture of local, State and Federal governments and universities.

This document is disseminated in the interest of information exchange. The contents of this report reflect the views of the authors who are responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California, the Federal Highway Administration, or the University of California. This publication does not constitute a standard, specification or regulation. This report does not constitute an endorsement of any product described herein.

For individuals with sensory disabilities, this document is available in Braille, large print, audiocassette, or compact disk. To obtain a copy of this document in one of these alternate formats, please contact: the Division of Research, Innovation and System Information, MS-83, California Department of Transportation, P.O. Box 942873, Sacramento, CA 94273-0001.

Advanced Highway Maintenance and Construction Technology Research Center

Department of Mechanical and Aerospace Engineering
University of California at Davis

CRASH ATTENUATOR DATA COLLECTION AND LIFE CYCLE TOOL DEVELOPMENT

Final Report

Bahram Ravani, Principal Investigator
and
George Burkett, Bradley Leong, and Duane Bennett

AHMCT Research Report: UCD- ARR-14-07-08-01
Final Report of Contract: 65A0416, Task ID: 2206

June 14, 2014

California Department of Transportation

List of Acronyms and Abbreviations

Acronym	Definition
ADT	Annual Daily Traffic
A/D	Analog to Digital
AHMCT	Advanced Highway Maintenance and Construction Technology
ATIRC	Advanced Transportation Infrastructure Research Center
CAL-COST	Crash Attenuators Life-cycle COST
Caltrans	California Department of Transportation
DOT	Department of Transportation
DRISI	Caltrans Division of Research, Innovation, and System Information
DSS	Decision Support System
FHWA	Federal Highway Administration
FOV	Field of View
GUI	Graphical User Interface
hr	hour
IMMS	Integrated Maintenance Management System
IP	Internet Protocol
IR	Infra-Red
LED	Light Emitting Diode
MASH	Manual for Assessing Safety Hardware
MPH	Miles per Hour
MCR	MATLAB Compiler Runtime
NCHRP	National Cooperative Highway Research Program
No.	Number
OEM	Original Equipment Manufacturer
IO	Inputs/Outputs
PIF	Public Interest Finding
POE	Power Over Ethernet
ROI	Region of Interest
RUC	Road User Cost
TMA	Truck Mounted Attenuator
VDC	Volts Direct Current
VoIP	Voice over IP

Acknowledgments

The authors thank the California Department of Transportation for their support, in particular Vue Her, John Jewell, and Bob Meline from the Division of Research, Innovation, and System Information. The authors also thank all the members of the Technical Advisory Group for this study.

Table of Contents

List of Figures

List of Tables

Abstract

This research study was aimed at data collection and development of a decision support tool for life cycle cost assessment of crash attenuators. Assessing attenuator life cycle costs based on in-place expected costs and not just the initial cost enhances the selection and procurement process resulting in cost savings for Caltrans Maintenance. The research question addressed included: can a decision support tool be developed based on detailed data that would allow proper estimation of in-service or life cycle rather than the initial cost of crash attenuators? In addressing this research question, first data needs to be collected on actual impact frequency at each crash attenuator site and actual repair costs of each type of attenuators. Data on generic repair costs of different classes of crash attenuators have been gathered based on IMMS (Integrated Maintenance Management System) data and has been used in the form of default values in a decision support system developed as part of this research. These default values can be over ridden by the user based on user input of such estimates. In order to help the user in terms of estimating the impact frequency at a site, a post processor for the IMMS data base has been developed that can search the IMMS data base for a site and obtain the repair frequency as well as actual data on total repair costs. The repair frequency is, however, a lower bound on impact frequency because the crash attenuator at a site may have had other impacts that did not require repairs. This is specially the case for sites with crash attenuators that are less likely to be damaged from impacts. In order to get a handle on the number of impacts that would not require repairs (these are referred to here as nuisance hits), an integrated impact sensor and site monitoring system was developed as part of this research to collect data on such impacts. This system was installed in three different sites and data was collected on nuisance hits. Data from IMMS data base as well as from the three sites with sensors have been incorporated into a decision support system with the acronym “CAL-COST” (Crash Attenuator Life cycle Cost) developed in this study. Input for CAL-COST requires estimates of impact frequency, repair costs and access costs for a site. As output it provides data including the cost break-even point for different classes of crash attenuators that can be matched to crash attenuators available in the market. The CAL-COST decision support tool can also be used to evaluate different Crash Attenuators products at a specific site and their life cycle costs.

Executive Summary

Motivation

This research study was conducted to develop data and methods that can be used for life cycle cost analysis of crash attenuators for highway applications. Proper life cycle cost analysis for a crash attenuator is a function of the crash attenuator repair costs, repair frequency and site access costs at any given location of its installation. Determination of life cycle cost of a crash attenuator cannot be properly made just based on its unit cost but should also consider its repair costs and its repair frequency based on frequency of hits at a site. It should also take into account access costs at the site of its installation. This includes the cost of any potential lane closures as well as the level of maintenance crew needed to access and repair the attenuator. This research study had two main objectives:

a. Develop methods for data collection and collect data that can be used in assessing life cycle costs of crash attenuators for highway applications.

b. Develop a decision support tool that would allow proper estimation of in-service or life cycle rather than the initial cost of crash attenuators.

Meeting these objectives can result in data and methods that can be used for proper selection of crash attenuators based on their in-service or life cycle costs and therefore reduce overall maintenance costs for Caltrans. It will also improve the efficiency of operations and safety of California highways due to the potential for lower frequency of access to a crash attenuator site for repair.

Approach

The approach used for data collection in this research study had two components: one involved extracting historic data from IMMS (Integrated Maintenance Management System) and the second involved field data collection on impact frequency for identification of nuisance hits. These hits are those that do not require repairs since some commercially available crash attenuators are not damaged up to a certain threshold or may have the ability for self-adjustment as compared to those that get damaged in most hits. Identification of the level of tolerance for nuisance hits for a crash attenuator design is important in properly assessing its life cycle costs due to lack of the need for repair as compared to a design with lower level of tolerance for such hits.

Historic data on generic repair costs of different classes of crash attenuators can be gathered from IMMS. In order to enhance this process and provide for a more efficient method of extracting data, an IMMS post-processing software was developed as part of this research study. The IMMS Post-Processor can search the IMMS data base for a site and obtain the repair frequency as well as actual data on total repair costs. This provides a sense of the impact history at the location that can be used for life cycle cost analysis. It can also be combined with satellite

images of the site, for example using Google Earth or other similar visualization tools, to better understand some of the site access requirements. The results provide some of the input needed for life cycle cost analysis.

When a site is equipped with crash attenuators that are less likely to be damaged from some impacts, then the repair frequency is a lower bound on impact frequency. The data collection part of this research therefore was not just limited to historical data within the IMMS. The approach also involved collecting field data at different sites with crash attenuators. A unique impact sensing and monitoring system was designed and installed at three different sites and data was collected at these sites.

In developing the decision support tool for life cycle cost analysis of a crash attenuator, the approach involved making the cost analysis related to the installation site for the crash attenuator. This is because the impact frequency and access costs varies for different sites and these two play important roles in the overall life cycle costs of a crash attenuator installation. A life cycle cost analysis methodology developed by AHMCT [3] was then combined with the IMMS post processor, historic data from IMMS, and data from the impact sensing and monitoring systems (installed at the three different sites) into a software tool called "CAL-COST" (Crash Attenuator Life cycle Cost).

Results and Recommendations

This research study has resulted in data and tools that can be used to assess life cycle cost of crash attenuators used in highway applications. An impact sensing and crash monitoring system was developed in this research study and was used to collect data at three crash attenuator sites in California. This crash monitoring system proved to be effective in capturing crashes that did not result in repairable damage to crash attenuators. These types of crashes are here referred to as nuisance hits. Data on nuisance hits does not exist in IMMS database since no repairs are performed in such incidences. This data is, however, important in proper life cycle cost assessment of crash attenuators. Data captured at the impact sensing and monitoring system sites also provided other very useful information on the effects of geometric designs of the sites on driver behavior providing insights into improving designs that can enhance highway safety.

IMMS database includes historic repair data at different crash attenuator sites in California. Much of this data is extracted, in this research study, from the IMMS database and is combined with data on nuisance hits from crash monitoring sites established in this research study to provide the data support for crash attenuator life cycle costs assessment. Furthermore an IMMS post processor is developed in this research study that facilitates user interactions with the IMMS database to easily obtain repair history and cost as well as impact frequency for crashes that require repairs at a site. Using this post processor, a sense of impact frequency and repair costs including an estimate of access cost for a site can be established. This data can then be combined with data or estimates of data on nuisance hits to assess the life cycle cost of different types of attenuators for a site.

In order to facilitate the life cycle cost analysis, this research has also developed a decision support tool entitled CAL-COST (Crash Attenuator Life cycle COST) which integrates all the data gathered and provided a tool that only requires a small set of inputs from the user. The user only needs to provide an estimate of access costs and impact frequency at a site. Such estimates can be made initially using the IMMS post processor. CAL-CPOT can then be used and it will provide recommendations for an appropriate class or category of attenuators at a site. This allows flexibility for the user to consider different options in terms of commercially available attenuators within that category accommodating other considerations such as delivery schedules, vendor reliability and so forth in selecting a crash attenuator.

In cases when there will be limited data on impact frequency or the level of nuisance hits and so forth, CAL-COST can still be a very useful decision support tool since it can be used in simulation mode. In this mode the user can simulate different options based on certain assumption on input parameters are not known a priori generating a list of options that can then be refined. In addition CAL-COST can plot comparison data on the cost of different crash attenuators at a site and determine the life required for breakeven points in terms of their comparative costs.

Based on the data collected in this research, the development of the decision support tool, and the experience gained in assessing life cycle costs of attenuators, one can make the following recommendations:

1. Selecting crash attenuators based on their life cycle cost rather than initial procurement cost can lead to more cost effective and leaner operations.

2. Crash attenuator selection process if divided into two steps: first selecting a proper category or class of attenuator and then selecting a product would allow for other non-technical considerations such as delivery schedules or other local requirements resulting in meeting both objectives of leaner operations as well as fulfilling other requirements.

3. Collecting and maintaining data on repair costs, impact frequency, nuisance hits, and access costs are highly recommended for all crash attenuator sites since such data can enhance future selection of appropriate crash attenuators.

4. Further develop the impact sensing and monitoring system so that it could provide notification to maintenance about attenuators which were impacted. This would facilitate the planning and scheduling of repair of the attenuators in a timely fashion improving highway safety. Addition of the notification function to the impact sensing and monitoring system would allow for event triggered maintenance and would significantly improve timely repairs of impacted crash attenuators enhancing highway safety.

5. Initially in order to minimize the number of crash monitoring sites, it is recommended to sort through IMMS data to identify “Hot Spots” as locations for the first phase of future installation of the impact sensing, monitoring, and notification system.

6. Providing the data from the crash sensing and monitoring system through a web site can allow manufacturers to improve their products benefiting the end users such as Caltrans.

7. Some attenuator products are very expensive to repair for low speed (light) impacts. These are not cost effective for Caltrans to deploy compared to other products that would require little if any cost to repair. The IMMS database provides data on the costs to repair products, but does not have any data on the type of impact which caused the damage. The impact sensing and monitoring system provides a video of the actual impacts to the attenuators. Such videos allow an evaluation of the robustness of attenuator products. Identifying and reducing the number of attenuators that are expensive to repair for low speed or light impacts are highly recommended since this can significantly reduce maintenance costs.

8. Developing plans to include GPS data instead of post mile information in IMMS for all repairs performed is recommended. This would allow better identification of exact crash attenuator sites as well as enhancing the IMMS database.

9. In its current form, the CAL-COST decision support tool can be primarily used for training/simulation. It is recommended to get the software tool out in the field so that work can be done to refine the data in its database. Once the initial wave of refinement is incorporated, the decision support tool would be able to quickly identify the optimal site specific products.

10. A higher resolution camera could be developed for the attenuator sensor system to record impacting vehicle license plate and driver identification to enable cost recovery for Caltrans cost of repairs.

11. Attenuator impact sensing notification could be transmitted to Caltrans traffic operations centers to improve mitigation and emergency response times.

Introduction

Crash attenuators are installed along travel ways to help protect the motoring public. They are designed to safely decelerate or redirect a vehicle with minimal risk to the impacting vehicles' occupants and other traveling public. In order for an attenuator to be accepted by the Federal Highway Administration (FHWA) for federal fund eligibility, it must meet minimum crash testing standards. These standards are outlined in the NCHRP Report 350 [1] which was released in 1992 and has been considered as the standard for some time. The NCHRP standards are being updated through the implementation of AASHTO MASH [2]. The tests outlined by these reports generally specify the vehicle weight, speed, trajectory, and the vehicle impact point. The collected results focus on the vehicle ride-down characteristics and occupant safety metrics, with little regard to the state of the attenuator after the impact. Chapter seven of the NCHRP 350 report mentions the need for in-service evaluations, but the industry in general has been slow to respond. One reason for this may be that repair costs fluctuate significantly depending on the type of impact. The required level of restoration can also be very subjective. Therefore, it would be impractical to presume that a representative repair cost figure could be determined from a small number of in-field case studies or the standard regiment of crash tests. Regardless, dramatic differences in service repair costs are implicitly linked to attenuator design characteristics.

A new class of severe duty attenuators is emerging. These products are designed to be easier to reset on the highway. However, these systems typically have higher initial costs. The cost effectiveness of this class of attenuator is overlooked with the current initial cost centric method. The AHMCT researchers have developed a methodology and a formula for assessing appropriate life cycle cost of crash attenuators [3]. This methodology and formula are based on detailed cost estimates of many items and parameters that require adequate data collection so that the methodology and the formula can be properly applied. This research addresses data collection and the development of a decision support tool. This information can be used for assessment of the in-service life cycle costs of crash attenuators for any specific site and determining alternative choices for a given application.

Background and Previous Work

The design of crash attenuators have evolved over the years from the "sacrificial" water or sand-filled barrels that were disposed of after each crash to more modern "severe-duty," which include designs that are "resettable" and "self-restoring". Some of the most popular crash attenuator products have been categorized in the NCHRP report 205 [4].

Some of the past research indicates that the main cost of crash attenuators are installation and repair costs [3, 5]. In addition, one needs to consider access costs for a site in determining the total cost of installations. Access costs can vary due to site geometry, traffic volume, and closure requirements and needs to be evaluated for each site. In evaluating the life cycle costs, however, it is probably more important to be able to estimate frequency of impacts requiring repairs. There seems to be very little work that takes into account a site's accident history [5-6].

The Kansas Department of Transportation and the FHWA had considered choosing the Smart [7] crash attenuator (over the Quadguard [8] attenuator) through a cost-predictive approach [6]. They correlated Annual Daily Traffic (ADT) with accidents per year and combined the ADT data with cost estimates. This resulted in a linear relationship of cost versus ADT for each attenuator type. This linear curve was then used to determine a threshold ADT past which one kind of attenuator would be more cost-effective than the other. This approach shows promise in situations where the initial high cost of an attenuator is difficult to justify without a thorough life-cycle cost evaluation, which is often the case with "severe duty" crash attenuators. This work from Kansas, however, did not have any indication of the statistical error present in the prediction and it can be seen that with their data, ADT and accidents per year are not significantly correlated. It is important to verify whether a model fits the data before applying it.

In 1998, the University of Maryland published a report for the state's department of transportation detailing a method for calculating the total life-cycle cost of a crash attenuator [5]. This study surveyed all fifty states and received responses from twenty-four of them. The survey was mainly focused on repair costs and frequency. The data from the survey was used to construct plots of yearly costs versus accidents per year. This is a straight-forward method of predicting costs, but the analyses were dependent on a single, crude average of accidents per year. One of the recommendations of this study was that DOT's should collect more data on repair costs for varying degrees of damage, attenuator performance for varying accident severities, and frequency of vehicle impacts.

The data collected are needed for input to a decision support tool that can be utilized in life cycle cost estimation for crash attenuators. Although, there have been several studies from different States Department of Transportation (DOT's) (see, for example, [9-11]) on performance evaluation of certain types of crash attenuators, the first work on developing a comprehensive decision support system for crash attenuators seems to be that of Spainhour, L. et al. [12]. This system is a multi-criteria decision support system using 29 criterions ranging from "tendency to spear a vehicle" to "repair costs" and "installation time". The system uses a data base of accident histories for each site. This system is very comprehensive and requires detailed historical data for each potential site. Its use was evaluated in the Master Degree thesis of J. Roth [13] and it was

concluded that the system is insensitive to the input site characteristics and more sensitive to the user input weights. The system, however, requires many user inputs making its usability limited. In addition there is often difficulty in obtaining the detailed data that is needed to effectively populate its data base.

In the present study, a decision support tool with the acronym "CAL-COST" (Crash Attenuator Life cycle Cost) has been developed that requires a much smaller set of inputs from the user. This tool builds upon a methodology that was developed earlier by AHMCT researchers [3]. The user can only specify estimates of access cost and impact frequency for a site. The system has default values for repair costs which can also be input by the user. Furthermore, the decision support tool specifies the class or category of crash attenuators for a site based on these user inputs allowing flexibility for the user to use other considerations such as delivery schedule, supplier reliability and so forth in selecting the appropriate crash attenuator within the category. In this study an integrated impact sensing and monitoring system has also been designed and three crash attenuator sites have been instrumented with such sensing and monitoring system. These installations have been used for detailed data collection that can enhance the ability of the users to obtain more accurate estimates of impact frequency as well as the nature of impacts at these sites. In addition, a postprocessor has been developed as part of this tool that can search the Caltrans IMMS data base to obtain estimates of impact frequency and actual repair costs. Data extracted from the IMMS data base using this postprocessor is combined with the field data collected from the three sites with integrated impact sensing and monitoring have been used as the initial data set for CAL-COST. The IMMS post-processor is discussed in the next section followed by sections on the design of the impact sensing and monitoring system and data collection sites. Finally, the CAL-COST decision support tool is discussed.

IMMS Post-Processor

The typical process for having an attenuator installed is to send the project out to bid, in which case, the lowest bidder is awarded the project. This often results in choosing the product with the lowest initial cost. The drawback is that in some cases, the attenuator, which is the cheapest initially, may be more expensive to repair after an impact or it may require more repairs after even low magnitude impacts increasing the life cycle cost of the system. Typically, there are many factors influencing attenuator performance which are highly dependent on the specific site. Furthermore, the number of impacts that the device will be subjected to is controlled, among other factors, by specific site conditions. In addition, some locations may be so difficult to access that the access cost can also be a critical piece of the puzzle. The decision support tool CAL-COST developed in this study addresses all these issues.

The IMMS Post-Processor was developed to assist the user in sorting through the Caltrans IMMS database to obtain some estimates of the data needed to be input to CAL-COST decision support tool. Integration of this Post-Processor with the decision support tool has enhanced the utility and functionality of CAL-COST. The IMMS Post-Processor allows the user to obtain repair history and cost as well as frequency of impacts that needed repairs at a site. The limitation to this is that the IMMS database has no information regarding nuisance hits. However, this process can at least give the user some initial insight to the site's history as well as repair frequency which is a lower bound on impact frequency.

Using the IMMS Post-Processor, a sense of the impact history at the location can be determined. This can be done by using the post mile and asset code as filters in the IMMS Post-Processor. As an example, a site for a crash attenuator in the Sacramento area was selected. The asset code used for this site was "03-SAC-050". The post-mile used was 2 miles ± 0.12 mile. Using this search, 34 repairs were identified as shown in Table 1 which occurred between November 13, 2001 and November 24, 2009. This data suggests that the average impact frequency requiring repairs at this location is once every 2.8 months. However, one difficulty associated with this location in particular is that there is a second attenuator in close proximity to this attenuator with probably the same post mile designation. This means that based solely on IMMS information, it is difficult to distinguish these two sites. The proximate attenuator to this site is a "Great" system. Based on this, 6 repairs can be eliminated from this site (based on IMMS comments). An additional repair from the list can be eliminated as it was a wall repair and not an attenuator repair. The impacts that are easily eliminated are indicated in bold text in Table 1. This brings the number of repairs down to 27 which means there is an impact requiring repairs once every 3.5 months. However, some of the repairs do not show a clear distinction between the two locations, and it is believed that the average time between repairs is greater than is suggested by Table 1. It should also be pointed out that the impacts associated with a specific site may have occurred on a product which has been replaced.

Table 1: Repair History at a Crash Attenuator Site (based on IMMS entries).

Wono	IMMS COMMENTS	Date	To Mile	From Mile	Total Cost	Approx. Time
12676	Replaced cartridges and r- sign	11/13/2001	2	2	$5,805.25	4.00 hr(s)
14123	Pull out attenuator, replace 1 R-sign, weld position chain (no cartridge replacement)	11/20/2001	2	2	$933	2.00 hr(s)
67850	NO COMMENT IN IMMS	5/16/2002	2	2	$1,009.06	4.00 hr(s)
80014	NO COMMENT IN IMMS	6/10/2002	2	2	$1,363.95	4.00 hr(s)
87741	NO COMMENT IN IMMS	6/28/2002	2	2	$3,998.18	4.00 hr(s)
103004	E/B L 50 PM 2.1 NO MANS LAND. ATTENUATOR DAMAGED, REPAIRED. WAITING FOR CHP REPORT.	7/30/2002	2.1	2.1	$913.35	4.00 hr(s)
168547	Nose knocked off and system collapsed; pull out and replace 4 cartridges & r sign	11/14/2002	2	2	$2,768.55	4.00 hr(s)
325263	NO COMMENT IN IMMS	6/13/2003	2.1	2.1	$583.67	2.33 hr(s)
334857	**Great system damaged noticed 6/29/03**	**6/30/2003**	**2**	**2**	**$7,978.94**	**7.67 hr(s)**
373094	Attenuator hit E/B 50 to S/B 99	8/18/2003	2.1	2.1	$445.27	4.00 hr(s)
432260	**Grate system hit 11/10/03 2052hrs; replaced 10 cartridges rebuilt systems using salvage material. Some material charged out 11/13/03 when time was done.**	**11/12/2003**	**2**	**2**	**$9,103.16**	**5.33 hr(s)**
542769	Pull out & clean up attn. 1 R-sign	4/13/2004	2	2	$308.42	2.00 hr(s)
552939	Pull out clean up replace 1 side panel & 1 R sign	4/27/2004	2.1	2.1	$1,366.18	1.80 hr(s)
619749	Replaced R-sign	7/28/2004	2	2	$92.24	1.00 hr(s)
715760	**Attenuator hit great system totaled replace 9 of 10 cartridges**	**12/15/2004**	**2**	**2**	**$9,219.90**	**3.60 hr(s)**
751934	Repair attenuator: replaced cartridges and used salvage nose piece and door panels. Additional damage to aluminum tube rail, left rail left connection.	2/3/2005	2.1	2.1	$7,503.83	6.50 hr(s)
829354	Noticed damage 5/25/05 to the nose cone, R-sign and cartridge	5/31/2005	2	2	$1,523.05	5.00 hr(s)
851764	**Damaged grate system. Job completed.**	**6/29/2005**	**2**	**2**	**$395.66**	**2.00 hr(s)**
879445	**Great system demolished replaced 5 cartridges 1 nose cone 1 R-sign**	**8/8/2005**	**2**	**2**	**$5,355.79**	**7.75 hr(s)**
901642	**G-system destroyed; will piece together as best we can**	**9/13/2005**	**2**	**2**	**$8,800.48**	**9.00 hr(s)**
926127	Full complement of cartridges, nose wrap, R-sign.	10/20/2005	2	2	$10,134.94	7.17 hr(s)
1252958	NO COMMENT IN IMMS	1/3/2007	2.1	2.1	$627.69	2.67 hr(s)
1281883	Repaired new react 350 attenuator system; utilized recycled parts to repair. No material costs.	2/6/2007	2	2	$351.80	3.00 hr(s)
1281913	Driver lost control of vehicle, struck react 350 attenuator system.	2/7/2007	2	2	$605.42	3.00 hr(s)
1288959	Debbie at TMC called Ted about accident at 50/99/51 split, attenuator damage.	2/17/2007	2	2	$1,562.92	4.83 hr(s)
1355933	CREW PULLED SYSTEM OUT TO MEMORY AND ADJUSTED THE CABLES, REPLACED R-SIGN.	5/7/2007	2	2	$718.97	4.00 hr(s)
1514307	NO COMMENT IN IMMS	11/29/2007	2	2	$854.81	3.80 hr(s)
1530340	Job completed.	12/14/2007	2.1	2.1	$6,835.46	8.00 hr(s)
1555957	Job completed salvaged materials	1/18/2008	2.1	2.1	$1,127.41	4.00 hr(s)
1609734	**Driver lost control of vehicle, struck attenuator system.**	**3/27/2008**	**2.1**	**2.1**	**$565.59**	**2.00 hr(s)**
1750347	NO COMMENT IN IMMS	9/15/2008	2.1	2.1	$872.35	4.00 hr(s)
1950934	Pull out and repair damaged attenuator system.	6/11/2009	2.1	2.1	$2,130.91	2.00 hr(s)
1977884	Repaired head wall and installed anchor bolts to retain barrels to the wall.	7/22/2009	2	2	$2,510.54	9.00 hr(s)
2066702	Pulled out react 350 repaired cables and put clamps on.	11/24/2009	2	2	$599.86	2.00 hr(s)

It is important to point out that additional information could easily be extrapolated from the IMMS data. This will be explained using the data that was presented in Table 1. As mentioned above with this particular site, further investigation needs to be made in order to determine which impacts are relevant to the specific site and not related to a nearby site. The illustration presented here will assume that all the data is correlated to the crash attenuator REACT350 [20].

Looking at the data, two parameters are easy to compute. The average repair cost is $2,910.00 and the average time of repair is 4.2 hours. These numbers are not the same as the values that need to be input to the CAL-COST decision support system. The input value to CAL-COST for repair cost should not include any information in regards to access. Additionally, the time that is specified in the IMMS data base includes not only the repair time, but also the time to set up the required traffic control.

In order to determine the basic repair cost from the above data, a few assumptions related to traffic control for the site should be made. The first assumption is on how long it takes to set up and remove the traffic control for the repair at the site. Here it is assumed that the traffic control takes 1 hour for setup and removal time. During this time, three workers and two trucks are assumed to be needed. During the repair, however, it is assumed that only one worker and one truck will be needed for traffic control as the other truck will be tied directly for the repair. The additional assumption will be that the average cost of labor is $28/hr and the average cost of equipment is $8/hr. (these rough estimates are based on IMMS data). Based on information from the IMMS database, an estimate can be made of the fixed access cost and the cost of maintaining access. Based on this information, the average bare repair cost at the site can be determined. The fixed access cost can be calculated to be $100, and the cost of maintaining access is $36/hour. This computation is based on a simple access cost model. The equation below can be used to estimate the site independent repair cost.

$$\$_{\overline{Rcost}} = \$_{\overline{Repair}} - \$_{ACF} - \$_{ACM}\left(\overline{t_{rep}} - t_{setup}\right)$$

In this equation, $\$_{\overline{Rcost}}$ is the repair cost, $\$_{ACF}$ is the fixed access cost, $\$_{ACM}$ is the cost of maintaining access per hour, $\overline{t_{rep}}$ and t_{setup} are respectively, the time to repair and the set up time. Using the numbers that have been presented, this comes out to the following:

$$\$_{\overline{Rcost}} = \$2{,}910.2 - \$100 - \$36/hr(4.2hr. - 1hr) = \$2{,}695/repair$$

Generally speaking, looking at all the data in the table presented above, the access cost is a relatively small percentage of the total cost. However, if one applies the same process to a specific work order number, such as work order number 2066702, which has a total repair cost of $599.86 and a two hour repair time, the bare repair cost comes out to be $463.86.

The goal of this example is to show that by using the IMMS Post-Processor, a person familiar with the site can easily begin to extract meaningful data from the IMMS database. Additionally, the back calculated bare repair cost can then be integrated into the CAL-COST resource file to help populate the data in order to help refine the software's default parameters to be more representative of the actual incurred costs. More details about the IMMS Post-Processor are provided in Appendix A.

Integrated Impact Sensing and Monitoring System

In an effort to understand the average cost of repair, it was important to gain an understanding of nuisance hits. These are impacts which occur on the highway that go unnoticed by maintenance due to the fact that they do not require repairs. Many of these impacts are significant enough that a repair would be warranted on a non-severe duty product. This represents a significant cost savings to a state DOT as well as a significant reduction in worker exposure. The only way to quantify the number of nuisance hits associated with a product is to implement a sensor system.

Sensors are required to determine an accurate value for the number of actual impacts since the IMMS data can only identify the number of impacts that required repairs. It should also be pointed out that different crash attenuator products have different nuisance thresholds which affect the denominator in the average cost of repair calculation. The sensor must be able to definitively capture impacts. This section outlines some of key sensor system components and design considerations for the system developed in this research study. Emphasis was placed on keeping the cost of the system low by using as many off the shelf components as possible. Three initial sites were instrumented (to be discussed later). The final sensor design will be presented below. The impact sensing and monitoring system is designed such that it can be easily modified for other applications in traffic operations. The design is an impact event triggered by the sensor and monitored by a video capture system that can be remotely accessed.

Background

Early on in the project, efforts were made to look for a commercially available sensor system for sensing nuisance hits to a crash attenuator. There were two slightly different systems that were developed by Energy Absorption which were investigated.

Evaluation of some of the commercially designed or available sensors indicated that some were designed with accelerometers and therefore suffered from false triggering due to vibrations resulting from vehicles traveling in close proximity of the crash attenuator. Other existing designs consisted of crash counters that were not electronic and wireless and therefore would require travel to the attenuator site on a periodic basis to count the number of nuisance hits.

Therefore, it was felt that developing a system which does not directly interact with the functional attenuator hardware would be the most appropriate. In developing a new sensor system for this purpose, it was important to first determine the kind of data to be captured. This is discussed next.

What is Being Captured?

The use of CAL-COST tool relies on having an accurate understanding of the average cost of repair. If only IMMS records are used, the average cost of repair is not corrected for those products which have a significantly higher nuisance threshold. Equations (1) and (2) below clarify this point mathematically:

$$\overline{\$_{rep(act)}} = \frac{\sum_{i=1}^{n_{repairs}} \$_{rep_i}}{m_{impacts}} \neq \overline{\$_{rep(imms)}} = \frac{\sum_{i=1}^{n_{repairs}} \$_{rep_i}}{n_{repairs}}, \quad (1)$$

$$\overline{\$_{rep(act)}} = \frac{\sum_{i=1}^{n_{repairs}} \$_{rep_i}}{m_{impacts}}, \quad (2)$$

where $\overline{\$_{rep(act)}}$ is the actual average cost of repairs, $\$_{rep_i}$ is the costs of a specific repair, $n_{repairs}$ is the total number of repairs, $m_{impacts}$ is the total number of impacts, and $\overline{\$_{rep(imms)}}$ is the average cost of repairs according to IMMS data.

This is meant to equalize all products by how they respond to typical impacts which means that all impacts must be factored in, not just those which require repairs. It should be important not to unfairly penalize those products which require trivial repairs for a majority of impacts and major repairs for a select few impacts. For some products, a major repair should be considered more of a cumulative repair than the repair caused by a single impact.

After identifying the different types of data that could be collected, it was felt that getting a video feed of the impact would provide the most conclusive data. This would allow for positively identifying any impact. Functionally, all that would be required of the data is to allow users to look at the video just before the impact as well as just after. This would allow for distinction between impacts and false triggers. A significant amount of additional information could be revealed by the data as well. The decision to capture video data created two key technical requirements for the camera system.

One technical requirement was that some kind of trigger needs to be in place in order to be notified of a potential impact. Continuously streaming video of a specific location would represent a significant amount of data. This data would not only take a lot of time to transfer, but would also be labor intensive to sort through. Capturing a smaller amount of video surrounding a triggering event would help put the impact into context and reduce both the amount of time required to sort the data and any system memory requirements.

Finally, another key technical requirement is associated with lighting. Typical cameras require visible light in order to record video. Ideally, the sensor system would be able to function 24 hours a day. One possible solution would be to place sufficient lighting at the specific deployment locations. However, placing a large floodlight at the attenuator's location would impact the site's properties in a way which would affect the impact frequency and hence have a direct effect on the quantity being measured. Therefore, having a camera which is capable of capturing both night and day video is the optimal solution. The selection and testing of a camera system and the complete design of a camera integrated sensor system is described in more detail in Appendix B.

Once the impact sensing and monitoring system was designed, the next step was selection of sites for installation of the system and data collection. This is described next.

Site Selection for Data Collection

This section describes the site selection for installation of the integrated impact sensing and monitoring system for data collection on impact frequency and identification of nuisance hits. During the site selection process, there were some key considerations that affected the mounting. The initial data collection was focused on sites with severe duty products. This significantly limited the list of viable locations. The next step was to look at the availability of power at the site. Although the design of a solar powered system was developed (see Appendix C), it was felt that having a direct power system would be more reliable and require significantly less development time. Since this is a developmental system, preference was made to locations which were within the proximity of the AHMCT research center. This helps in system development simply due to the shorter travel time required to get to the instrumented sites. The sites should be in areas which experience impacts somewhat frequently. Based on these considerations, three sites in the Sacramento area where identified that meet all the above criteria.

Site A: React 350 at the Highway 50 and Highway 80 Split

General

The first site has a React 350. The site as depicted in Figure 1. was chosen after speaking with Caltrans maintenance personnel who said that the attenuator gets hit fairly regularly.

Figure 1. Site A.

Directly behind the attenuator at this site is a concrete wall which can be seen Figure 2. The sensor system could easily be mounted and placed within the confines of the concrete wall.

Figure 2: Backside of REACT 350 attenuator.

The guardrail that can be seen in Figure 2 above has bolts that extend through concrete into the cutout area. These bolts are illustrated below in Figure 3.

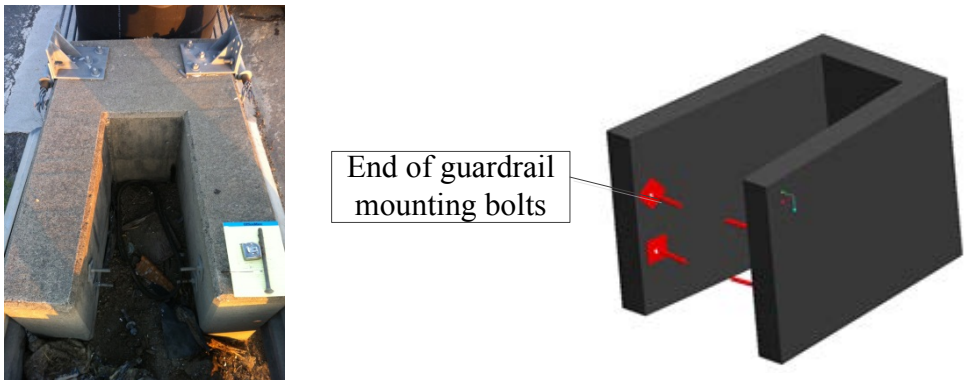

Figure 3: Back of attenuator detail.

Placing the mast within the area shown in Figure 3 seemed like an ideal situation. No modifications need to be made to the exising structure. The concrete will provide some protection to the sensor system from any impact.

Installation

The guardrail bolts shown in Figure 3 above served as a mounting point for the mast support at this location. The basic mast support design consists of four angle brackets that are free to pivot relative to the main mounting channel. This compensated for the taper in the concrete wall. Also, one pair of the angle brackets attach through a slotted mounting system to the main body of the mounting system. This corrected for any errors in the spacing between the sides of the concrete wall.

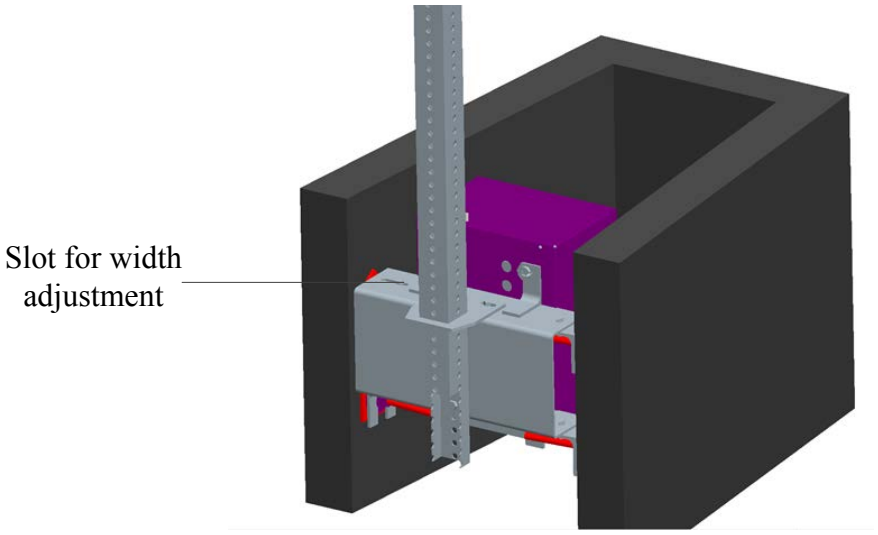

Figure 4: Basic sensor mounting.

Mounting the mast at this site was fairly straightforward. There were just some issues only due to access to electricity but these issues were easily resolved.

Camera Field of View

The camera's field of view is important in order to reduce the number of false positives while still providing sufficient data. With that said, given that the motion zones within the field of view can easily be modified, there is no need to precisely position the cameras. The field of view for this site is presented below in Figure 5, which shows the camera's field of view during the day. Figure 6 presents a nighttime view of the site. One thing to note is that due to the IR illuminator, there are points within the field of view which are saturated with light. These zones will be significantly less affected by headlight triggering.

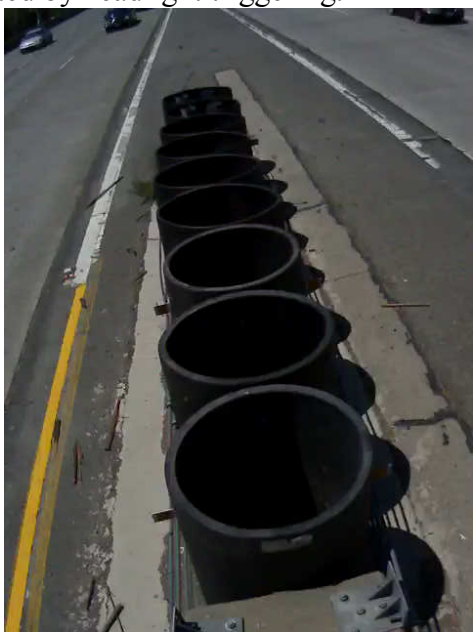

Figure 5: Daytime view of REACT350 attenuator.

Figure 6: REACT350 night-time view.

Next, it is useful to look at the motion zones. Given that the barrels of a REACT 350 attenuator are connected with cables, it is believed that both a side and front impact will transfer enough motion throughout the attenuator to cause sufficient displacement. The motion zones also are going to be centered on areas that are well illuminated by the IR emitter at night in order to reduce headlight triggering. This is illustrated in Figure 7 below.

Figure 7: REACT350 motion zones.

Site B: Smart Cushion at the Highway 99 South Split with Highway 50 East/Highway 80 West

General

The second site that was chosen was a Smart Cushion located at the split between 99 south and 50 East/80 West. A general view of this site can be seen in Figure 8. The biggest challenge here was contending with the sign behind the attenuator. Ideally, it would be best to place the sensor upstream from the sign in order to get a better view of the attenuator. However, after talking with our Caltrans project managers, the camera system could not interfere with the signs visibility. Hence a different approach was taken. The mast mounting at this site was designed to connect to the existing bolts which are used to support the existing sign.

Figure 8: General View of Smart Cushion Site.

Looking closely at Figure 9, the reader will appreciate how close the system is to power compared to the React site. This meant that any concerns in regards to voltage drop between the power box and the sensor box were insignificant if the same gauge wire used for the React site was also used here.

Figure 9: Rear View of the Smart Cushion Site.

Repair History

Similar to the REACT site above, the IMMS tool was used to help identify the repair history at this location. For this site, the asset code of 03-SAC-051 was used along with the post mile information of 0.1 mile ±0.1. This search aided in identifying the repairs given in Table 2: Impact history at Smart Cushion location. This shows that between May 16, 2002 and February 23, 2009, there were a total of 12 repairs. This data suggests that this attenuator gets hit once every 7.5 months.

Table 2: Impact history at Smart Cushion location.

Wono.	IMMS COMMENTS	Date	To Mile	From Mile	Total Cost
67704	NO COMMENT IN IMMS	5/16/2002	0.1	0.1	$319.20
210563	Called for attn. SB51 to E/W 50 conn. Closed doors; will reset Tue. Morning	1/18/2003	0.1	0.1	$3,512.10
499826	Reset attenuator; replaced R-sign; Material charged out following day	2/17/2004	0.1	0.1	$159.76
581852	Pull out, clean & paint replace R-sign	6/7/2004	0.1	0.1	$280.14
1003128	Sand barrel hit, completed	2/2/2006	0.1	0.1	$2,707.88
1060784	NO COMMENT IN IMMS	4/24/2006	0.1	0.1	$909.87
1327664	04/02/07 Bill and Ted checked Attenuator damage.,,04/04/07 Crew worked to replace all damaged inserts, clean up the area.	4/2/2007	0.1	0.1	$7,850.42
1332794	04/09/07 Crew had to replace 8 inserts. Clean the area.	4/9/2007	0.1	0.1	$6,271.79
1404543	07/09/07 crew replaced damaged hex foam inserts, replaced 30' X 1" support cable. cleaned area.	7/5/2007	0.09	0.09	$5,862.49
1770506	10-18 responded to TMC call out for damaged attenuator due to accident.	10/18/2008	0.1	0.1	$2,372.53
1969622	Damaged attenuator system. Replaced cartridges and diaphram	7/6/2009	0.1	0.1	$3,896.99
2134878	Replaced cartridges, diaphrams and chevron sign. rebuilt system	2/23/2010	0.09	0.09	$4,603.56

Installation

As mentioned above, the mast was mounted just behind the sign post. Given the height of the sign, it was decided that the camera and IR illuminator should be mounted just below the sign. The key concern here is that the camera system cannot interfere with the sign visibility. A general view of this installation is presented in Figure 10.

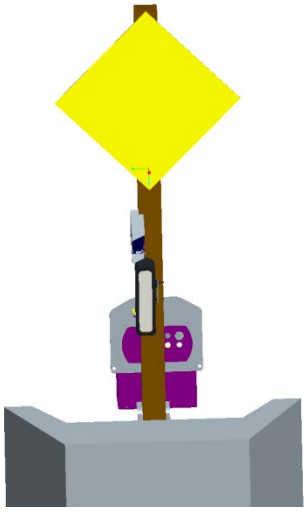

Figure 10: Over view of Smart Cushion installation.

Figure 11 depicts an overview of how the mast is mounted. The lower bracket pins the base of the telescoping tube. The upper bracket has a square socket that allows the tube to slide in, but is tight enough to reduce vibration. This system allows the tube to be easily mounted without having detailed information in regards to the bolt spacing.

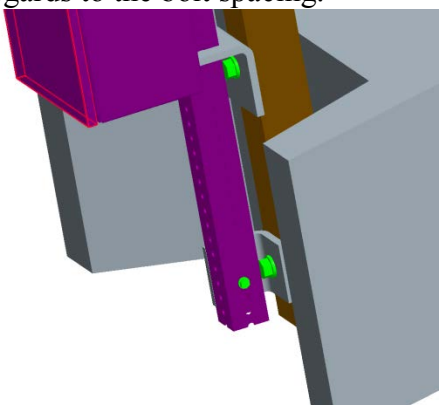

Figure 11: Detailed look at mast mounting brackets.

Camera Field of View

Out of the three sites, the Smart Cushion installation has the worst field of view. The key issue with this installation had to do with the mounting system. This site was ideal for power availability. However, because of the sign at the site, the mounting was forced to be significantly behind the attenuator. This view can be seen in Figure 12.

Figure 12 Daytime view of Smart Cushion installation

As can been seen, this perspective gives the camera a very flat point of view which causes the system to be more effected by headlights of oncoming traffic at night. This is easier to see by looking at the nighttime field of view for this site as shown in Figure 13.

Figure 13: Nighttime view of the Smart Cushion installation.

One issue with this site that still needs to be resolved is switching the camera between normal imaging and IR imaging modes. There is a street light near this attenuator as well as lighting for signage above the attenuator. This means that finding the correct settings for the camera, which will force the camera to work in this manner, is difficult. This can be seen by the fact that the camera brightness and contrast settings are so low that the camera actually operates in night-mode during portions of the day and operates in day mode at night (which can be seen in Figure 14).

Defining the motion zones for this site are more challenging then the REACT 350. First, very little displacement will happen to the attenuator in a side impact. One good thing to note about this is that given the site specifics, common sense would dictate that a frontal impact would be more likely. This means that the motion zone should be focused more on the front of the attenuator. This can be seen in Figure 14 and Figure 15.

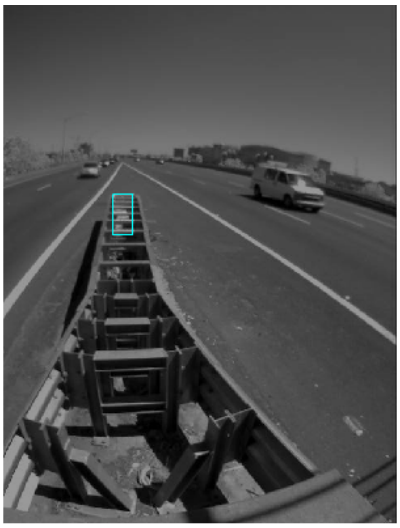

Figure 14: Daytime view of Smart cushion.

Figure 15: Nighttime view of the Smart cushion.

An additional issue with the motion zones associated for Smart Cushion is that the structure of the cushion is somewhat open. This means that the shadows in the attenuator change due to vehicles illuminating some of the internal structure from underneath. This means that any zone which shows significant depth within the attenuator will be a source of false triggers.

Site C: Compressor at Westbound Highway 80 and the 160 Split.

General

The compressor site also presented some challenges. One to note is that since the compressor is a relatively new product, there were very few compressor sites to choose from. Figure 16: Compressor site presents a basic view of the compressor site. The power pole for this site is located closer than the power at the React site so voltage drop was of no concern since the cable used between the power box and the sensor box was standardized for every installation.

Figure 16: Compressor site.

Repair History

Using the IMMS tool for this site also gave a sense of the impact history. The asset code for this location is 03-SAC-051 and using post mile values of 3.5 miles ± .51 mile, yielded 12 repairs from August 8, 2004 to September 21, 2009 as shown in Table 3. This shows that the attenuator gets hit approximately once every 5 months.

Table 3: Impact history at the Compressor site.

Wono.	IMMS Comments	Date	To Mile	From Mile	Cost
636120	All barrels replaced	8/23/2004	3.6	3.6	$3,008.96
892917	Job not completed on 8/29/05 will charge materials when job is completed on 8/30/05	8/29/2005	3.6	3.6	$2,475.43
935270	Inspect, clean, paint, & minor repaires	10/31/2005	3.6	3.6	$1,419.51
965425	Replace 1400# barrel	12/14/2005	3.6	3.6	$814.28
1165379	NO COMMENT IN IMMS	9/11/2006	3.6	3.6	$312.75
1311319	03/14/07 Crew replaced damaged barrel, corrected system layout. Cleaned area	3/14/2007	3.6	3.6	$565.71
1428556	08/07/2007 Crew replaced 1 1400 barrel and 3 lids and an R-sign. repositioned the system and cleaned up debris.	8/7/2007	3.6	3.6	$1,285.81
1878733	Accident happened 03/09/09 @ 1305 hrs. ncic9250 12602 off i d #	3/10/2009	4	4	$1,177.59
1920829	Damaged complete array of sand barrels. Also guardrail damaged.	5/4/2009	3.6	3.6	$3,500.18
1940931	NO COMMENT IN IMMS	5/30/2009	3.69	3.69	$625.67
1941672	Sand barrels destroyed.	6/1/2009	9	3.9	$3,209.46
2019509	Sand barrels damaged. Replaced 3	9/21/2009	4	4	$1,212.44

Installation

At this particular site, the only viable option to mounting the camera was to clamp the system to the back of the attenuator. The main issue with doing so is that the camera mount would not be fully isolated from the impact. In order to prevent large deflection in the camera mast, the support system was designed to attach at the base of the attenuator. This allows the camera to be more isolated from any large deflections that may occur at the top of the attenuator. The basic design can be seen in Figure 17.

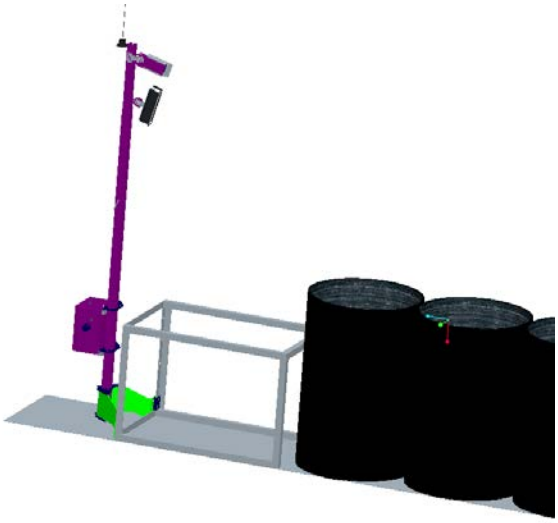

Figure 17: Overview of compressor mount.

The Figure 18 presents more detail on the mounting system. The system has two clamps that grip the rear frame on the attenuator. Additional support is provided to the system through self-leveling feet that are integrated into the design.

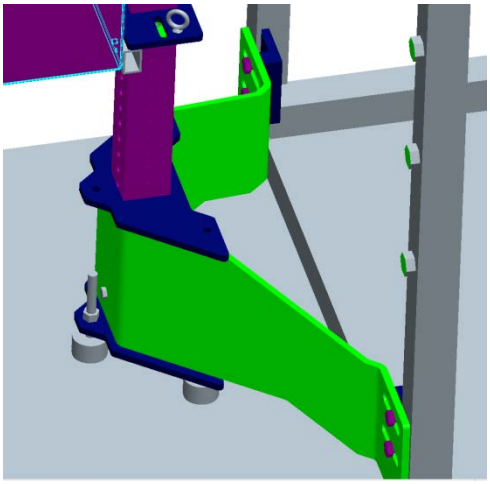

Figure 18: Base of compressor mounting.

The key to this concept was to stabilize the camera mount a in such a way as to minimize the amount of displacement transferred to the camera system from the attenuator in the event of an impact.

Camera Field of View

Out of the three sites, the Compressor installation has what is believed to be the best field of view. The camera is positioned so that it is not looking down on the attenuator as much. The headlights of oncoming cars will have little direct effect on the camera. The daytime and nighttime views are shown in Figure 19 and Figure 20 respectively.

Figure 19: Daytime view of the Compressor.

Figure 20: Nighttime view of the Compressor.

Motion zones associated with the site are very easy to identify. Similar to the React 350, the plastic sections of the Compressor are loosely connected. This is done through a guardrail like

structure that is connected to the attenuator. This means that any side impact will transfer some displacement throughout the attenuator. Also, given the camera's field of view, identifying zones which are saturated by the illuminator at night is very straightforward. This can be seen in Figure 21 and Figure 22.

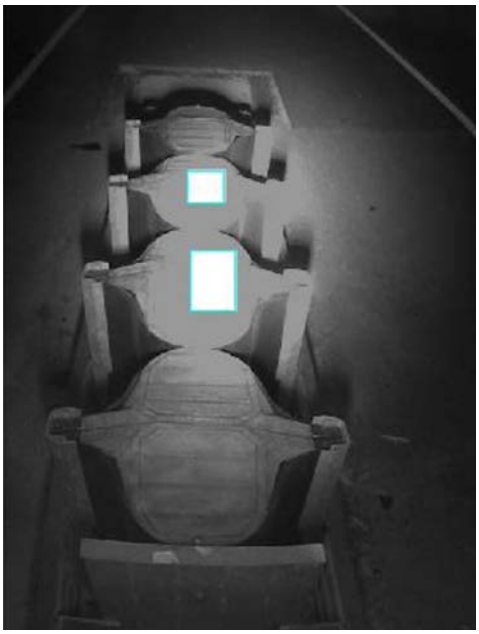

Figure 21: Nighttime view of motion zones.

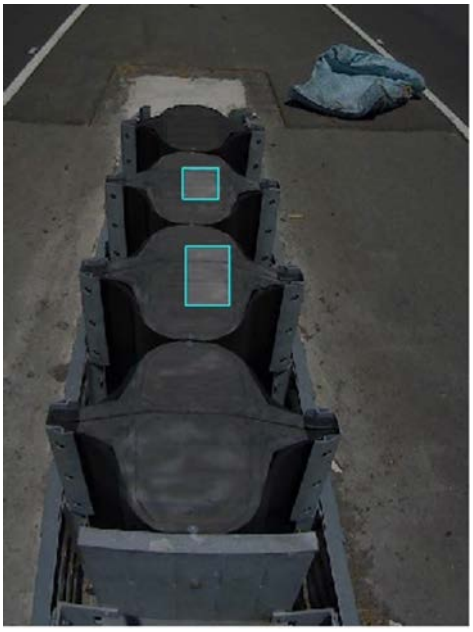

Figure 22: Daytime view of the Compressor.

Sensor Performance and Results

The sensor systems discussed above were installed on May 18, 2012. As of June 14, 2012 no impacts were identified. However, one near miss was recorded and can be seen in Figure 23. This event was captured because of the changes in light, which became a factor in this instance given the proximity of the car to the attenuator.

Figure 23: Near miss at REACT 350 site.

In general, the sensors have been reliable and have functioned well. The largest remaining issue is reducing the number of false positives for the systems. Once an impact is identified, the various camera settings can be further optimized in order to reduce the number of false triggers.

After initial installation, there was some troubleshooting in the field which was associated with powering the system. The difficulty here is that the small battery charger in the power box does not have sufficient capability to revitalize the battery when it drains to zero. This means that the instances where the battery died, two trips were often required in order to bring the battery back to the lab for a complete recharge and then go back to reinstall it. However, this should be somewhat expected as this is a sensor development project which is bound to have some small

issues. In general, given the overall complexity of the system, it has been reliable. The cellular modem system has functioned well. The power systems switch back and forth correctly. Access to the camera seems to be reliable both from a Smartphone and a PC. On occasions, there have been issues remotely accessing the camera; however, these issues have often remedied themselves by trying to access the cameras at a different time.

The impact frequency discussed above should be considered a maximum value. These particular products were instrumented because they have a high nuisance threshold. This would suggest that impacts happen more often than the data above suggests. However, this is speculation with no associated physical evidence which is why there is a need to instrument these sites. For all three sites, there are instances which demonstrate a large span between impacts. This can be quickly shown in the data as there is a year for all the sites which show no IMMS entries (2008 – Compressor site, 2005 – Smart Cushion site, and 2006 – REACT 350 site). This would suggest that the sensors are functioning properly and that there is no cause for immediate alarm. This is also supported by the fact that there has been no notification from CALTRANS in regards to these three locations.

Identified Traffic Hazard – I-80-Business Highway 160 Split

During the course of attenuator monitoring, a reoccurring traffic issue was exposed with the sensor device located along Westbound I80Bus and 160. Figure 24 shows a large truck moving across the camera's field of view. Looking at this from the direction of traffic gives a driver's perspective at the site as shown in Figure 25.

Figure 24: Truck crossing in front of the sensor.

Figure 25: Driver's perspective of site C issue.

Looking at this in detail, it became apparent that vehicles entering highway 160 from Arden Way southbound were dangerously crossing 2 lanes of Hwy 160 traffic to cut-over to I80BUS westbound. The green path in Figure 26 below indicates the designed route; the red path illustrates the hazard. This issue was not an isolated event but was routinely noticed. The image shown in Figure 24 was the most extreme as the truck in question was pulling doubles.

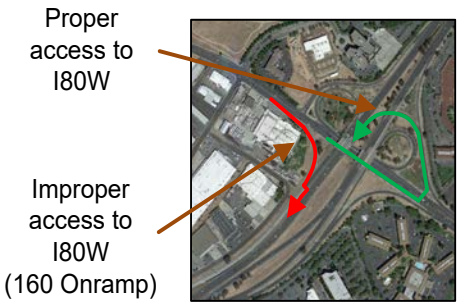

Figure 26: Overview of observed "site C" issue.

At this site, the sensor system was able to identify a clear risk to the general public. After presenting this issue to Caltrans Maintenance, a channelizing system was determined to be the best mitigation strategy. Plastic delineators were installed to further discourage the unsafe driving behavior as shown in Figure 27. The net result is a safer highway interchange. This change could quite possibly prevent a deadly accident from occurring in the future.

Figure 27: "Site C" with installed delineators.

This example shows that the sensor system may be used for other situations where simply observing driver behavior over a period of time may be helpful in order to understand the current traffic dynamics. Once a better understanding is achieved, methods for reducing the inherent risk at the site can be implemented in order to improve the overall safety of the site.

CAL-COST Tool

The CAL-COST tool was primarily developed as a Decision Support System (DDS) utilizing a life cycle based methodology developed earlier [3] for selection of crash attenuator category most appropriate for a site. This section is intended to illustrate the details of the tool. It should be pointed out, however, that this tool is presently a research demonstration package and not a robust production level software package.

The tool provides a framework for traffic designers to systematically approach the process of determining the best crash attenuator category based on the life cycle cost. It should be pointed out, however, that a decision support tool such as CAL-COST is not the best replacement for years of field experience, but never the less, it can provide assistance and a point for comparison of different options in selection of a crash attenuator for a site. It is designed to be a user-centric system which allows a user to modify the input values to those which may be more representative of how specific groups of attenuators perform in the field.

The output of the tool gives the user a clear sense of what category of product to use. Because there might be insufficient data on available products and their performance or since new products are introduced periodically, the tool is not designed to recommend a specific product but to provide a recommended category of products. This is because beyond looking at the class of attenuators, there are other factors which will affect the appropriateness of a specific product. One example of such factors is the physical size of the product, others include product availability and delivery schedule. A utility is also developed within the CAL-COST tool that would allow product selection based on such additional parameters. This utility is referred to as Product-Selection (PS) utility and is described in more detail later in this section.

CAL-COST System Requirements

The current version of the CAL-COST tool is based on the MATLAB environment. The tool can be compiled as a stand-alone application. In order to run the software in this fashion, the host computer must have the same version of the MATLAB MCR library that was used to compile the software. This allows the tool to be used outside the traditional MATLAB environment. Using a compiled version of the application facilitates having a very basic mechanism for version control.

The current version of the tool interacts with Microsoft Office (both Word and Excel). Therefore Office is required to be on the host machine for the program to properly execute. The Excel resource file may need to be saved to a different version of Office if the host computer is using an older version. The tool has a report generator that will automatically determine what version of Office is loaded on the machine and creates a report of the results in that version of word. Therefore the report generator does not place any Office version restrictions in regards to tool execution. The Office version checking system is merely put in place since some users may not have the most current version of Office on their computer. The tool was mainly tested using Office 2007, but has been used with newer versions.

The current version of the tool requires a minimal resolution of 1900 x 1200 pixels. This is due to the fact that the Graphical User Interface (GUI) was not written in a way that allows for resizing the window like other windows applications.

The current form of the tool is not intended for large scale distribution. CAL-COST tool is intended to serve as a developmental platform to aid in identifying how to improve the life cycle-cost analysis process and product deployment justification. However, it is felt that the tool is in a form which could be used to aid in the PIF (Public Interest Finding) submission process with little effort. However, part of this effort would help refine the default values used by the tool for future use.

Using CAL-COST

Fundamentally, CAL-COST can be utilized in two ways. One way CAL-COST can be used is to provide a recommended attenuator category with very little information by the user. Another way it can be used is to simulate the effects of changing parameters in order to understand the effects of specific factors. Examples of both uses will be presented below.

Category Recommendation Example

There is very little information that is required in order to have CAL-COST present a specific category recommendation. First, the user must have a rough idea about the traffic control required at the site; for example, say the location requires a lane closure. In order to setup and remove the lane closure, 3 workers are required with two trucks. Setting up the traffic control takes an hour, and removing the traffic control takes the same amount of time. One truck is an attenuator truck, while the other is a cone body with a driver and another operator to set the cones. Equipment is typically charged at a rate of $10/hr and labor costs approximately $25/hr. This means the fixed access cost would be $190 total. Once the traffic control is in place, the TMA truck and driver must help shield the worksite which means the cost to maintain safe working conditions is $35/hr. The user can then choose to estimate the public impact cost. For example, say there is minimal public impact, so the public impact cost is $100/hr. The user can then input these values into the traffic control input window as shown in Figure 28 below.

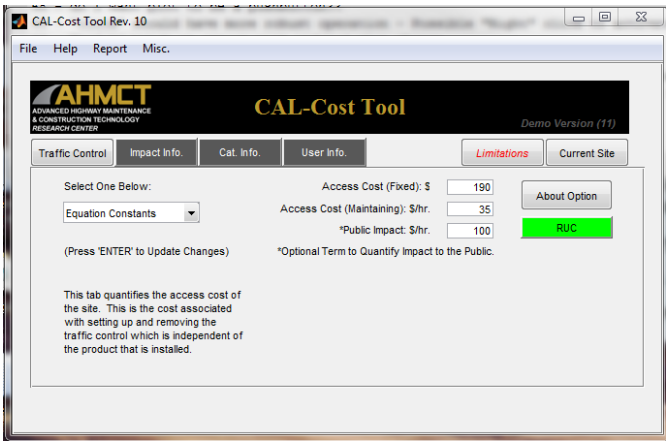

Figure 28: Traffic control for examples.

Once this data is entered, the user can then plot the cost relationships as shown in Figure 29. This plot gives a sense of how the various categories perform with respect to each other.
Figure 29 shows that a repairable attenuator is never the optimal category and that the repairable and sacrificial categories show very little difference in terms of slope.

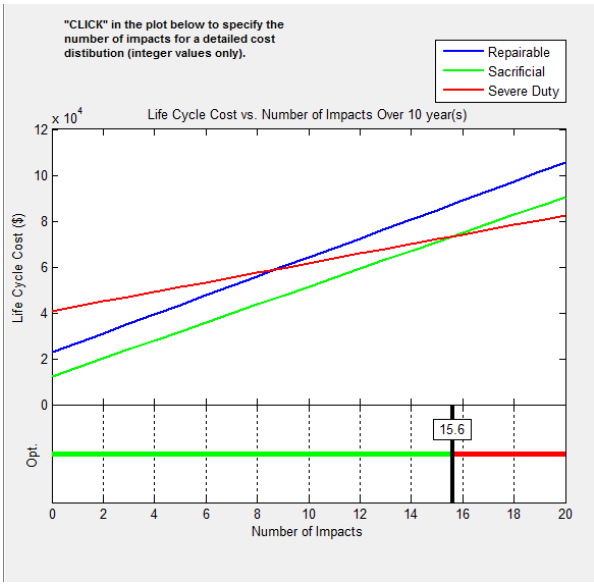

Figure 29: Plot of cost vs. Impacts.

The user can look at the default values in order to assess the default cost values. The default value for average repair cost for sacrificial attenuators seems to be relatively low compared to the installation cost ($12,279.52 and $3,193.31 respectively). The values that are integrated into the system are based on IMMS information. The goal was to identify people to work within Caltrans to help refine the data in a way that makes the default values much more representative than the current IMMS data. However, for the sake of an example, the default value will be changed to $6,139.76 which is half of the default installation cost which seems more reasonable given that the category of interest is sacrificial. Making this change yields the graph shown in Figure 30.

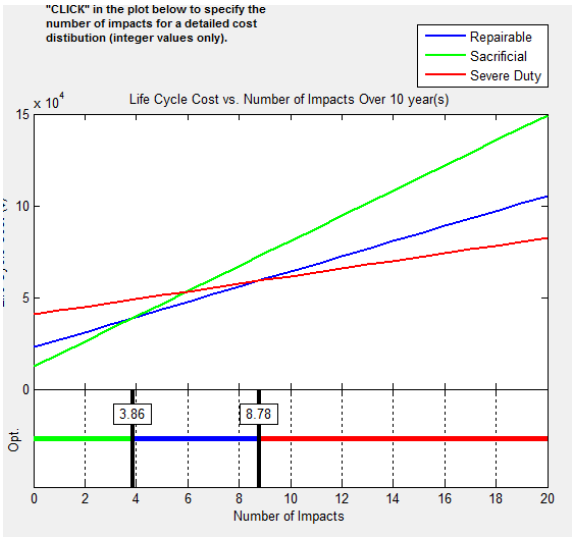

Figure 30: Plot of cost vs. Impact (with modified sacrificial repair cost).

The next part for category selection is that the site of interest is known to have an impact frequency of one impact every year and half. This means that over the course of 10 years, the site will experience 6 or 7 impacts. The software is set up for only integer number of impacts, so, some additional information will be presented by specifically looking at 7 impacts. An over-view of the results is presented in Figure 31 below.

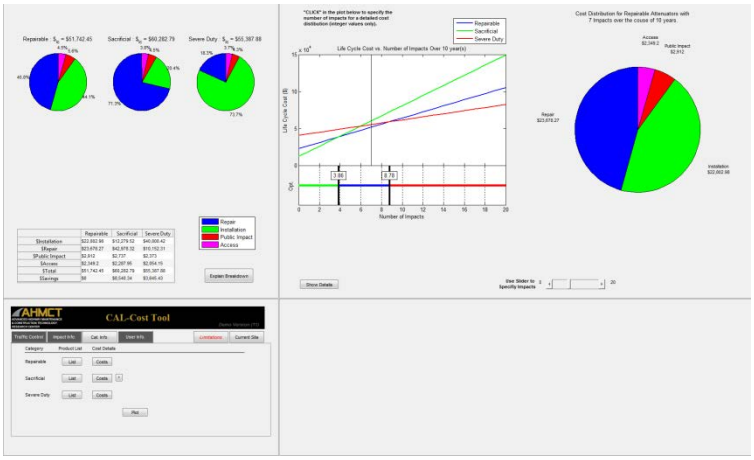

Figure 31: Over-view of category example results.

The pie chart to the right of the life cycle cost shows a breakdown of how the cost is distributed in order to give the user some additional insight into the problem and can be seen in more detail in Figure 32. One can see that in this case, a vast majority of the cost consists of installation and repair.

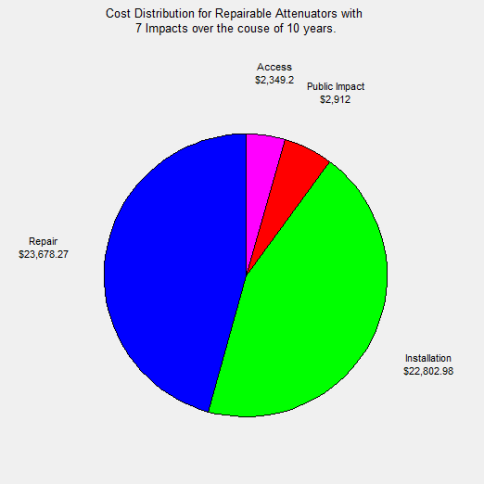

Figure 32: Cost distribution for a known example.

In this example, the repairable category would be the optimum selection. If the tool is further developed and the default values are better defined by experts in the field, the process of selecting the optimum category would be a very trivial process as has been shown in the above example. The user would only have to know access costs, public impact cost, and impact frequency of a site of interest in order to quickly obtain a recommended category from the CAL-COST tool.

Simulation Example

Another way in which CAL-COST can be used is to simulate the effects that various parameters can have on the category of crash attenuator that is being recommended. This can be done by changing the parameters systematically and looking at the results.

For example, one could ask, "What is the effect of public impact on the recommended category?" In order to simplify this example the access costs and the updated average cost of repair from the previous example will be used. The previous example showed that the crossover points were at 3.86 impacts and 8.78 impacts. Doubling the public impact changes the cross-over points to 3.9 and 8.46 impacts respectively. Next, as this number is changed to $400/hr again, the values shift to 3.97 impacts and 7.89 impacts. What this simulation is showing is that by increasing the public impact cost the attenuators which are typically faster to repair become more attractive, which is to be expected. The actual magnitude of the effect will become more accurate as the default category values are refined.

References

1. Ross, H. E. Jr., D. L. Sicking, R. A. Zimmer, and J. D. Miller, "Recommended Procedures for the Safety Performance Evaluation of Highway Features", NCHRP Report 350, Transportation Research Board, National Research Council, National Academy Press, Washington, D.C., 1993, 64 pages. <http://onlinepubs.trb.org/onlinepubs/nchrp/nchrp_rpt_350-a.pdf>.

2. "Manual for Assessing Safety Hardware (MASH)", American Association of State Highway and Transportation Officials, October 2009.

3. Burkett, G., D. A. Bennet, and B. Ravani, "Crash Attenuator Usage Along Travelways and in Workzones." *AHMCT* Report No. UCD-ARR-10-07-16-01, University of California-Davis, AHMCT Research Center, July 16, 2010.

4. Carney, J., "Performance and Operational Experience of Crash Cushions." NCHRP Report 205, Transportation Research Board, National Research Council, Washington, D.C., 1994, 87 pages.

5. Khorsandian, F., Schonfield, P. (1988). "Evaluation of Impact Attenuators ", MD-86/03. College Park, MD: University of Maryland, January 1988.

6. Kansas DOT/FHWA SCI Discussion. March 24, 2008.

7. SMART CUSHION INNOVATIONS, SCI Products Inc., Work Area Protection Corporation, 2005, 6 pages.

8. QuadGaurd System, "Product Manual", Energy Absorption Systems, Inc., 2008, 60 pages.

9. Mak, K., Sicking, D., "Continuous evaluation of in-service highway safety feature performance." *Volume 482 of Final report (Arizona Dept. of Transportation)*, 2002.

10. Horton, J. (2013). "Severe duty crash cushions: Experiences of California, Kansas, and Nevada." *NHI Innovations Web Conference*, January 17, 2013.

11. Ray, M.H., Weir, J., Hopp, J., "In-Service Performance of Traffic Barriers. " NCHRP Report 490, Transportation Research Board, National Research Council, Washington, D.C., 2003, <http://onlinepubs.trb.org/onlinepubs/nchrp/nchrp_rpt_490.pdf>.

12. Spainhour, L. K., P. V. Mtenga, and J. Sobanjo, "Multicriteria DSS with Historical Database for Attenuator Selection", J. of Computer in Civil Engineering, 199, 13: 187-197.

13. Roth, J., “Decision Support System to Rank and Evaluate Crash Attenuators”, M.S. Thesis, Department of Civil Engineering, The Florida State University, 2004, 120 pages.

Appendix A

IMMS Post-Processor

Background

One of the primary data sources for repair information and impact history is the Caltrans IMMS database. The CAL-COST utility has a subcomponent which can be used to interact with the IMMS data. This subcomponent is accessed through the IMMS button shown in Figure 86, or directly accessed from the MATLAB command prompt. The idea was to give the user a means to quickly sort through the IMMS data and get meaningful information about a site. In order to do this, a brief understanding of the IMMS data is required. After understanding the basics of the raw data, efforts can then be made to consolidate the information in a way that makes it more accessible to the user.

Before looking at the data in detail, the user must understand how the data is stored. The database structure will allow for data to be accessed in a systematic process. This is the fundamental property of databases which make them such a useful tool.

After understanding the data structure, efforts must be made to consolidate the information into a form that is the most relevant to the problem. In the information presented here, the consolidation metrics are focused on understanding the factors which are related to a single repair. Looking through the data can be difficult since a single work order number, which is synonymous with a single repair, may have multiple IMMS entries. The consolidation process helps reduce a work order number to a single line of information which simplifies data interpretation. This prevents the user from being bombarded with too much information at once while still having the ability to look at all the details.

After consolidating the data, the information can be filtered to only those pieces which are relevant to the main focus of this study (a specific site for example). This filtering process quickly narrows in on the most relevant subset of data. In this situation, the typical filtering process will involve either repairs associated with a single site, or a single product. This single product filtering could ultimately serve as a mechanism to pipeline IMMS data into the CAL-COST resource file.

The main drive behind this process is to facilitate interaction with the IMMS data. The starting point of the process that is implemented in the IMMS tool is an excel spreadsheet which contains the raw data. This type of file can be directly exported from the IMMS database. After developing a basic understanding of the structure of the database, efforts are made to consolidate the data. Once the data is consolidated, various filters further reduce the data to more specific information. The main idea here was to give the user the ability to interact with the data as easily as possible.

Structure of the IMMS database

The IMMS database is an internal Caltrans database used to organize work order information. Table 4 below lists out the fields of interest in the IMMS database as well as a brief explanation

of their meaning associated with a single item. The various types of information contained in these fields allow for the implementation of various filtering mechanisms for sorting the IMMS data. Many of the fields are fairly self-evident from the description provided in Table 4; however, some fields require a more detailed explanation.

Table 4 Description of IMMS fields.

IMMS Field	Description
Month	The month the work took place
Date	The complete date the work took place
Dist	The Caltrans district
Unit	Internal Caltrans designation
Activity	The is the Code used to specify the activity that is part of the IMMS database structure set up by CAL-trans
Activity Description	A text description of the activity
IMMS Asset	A code explaining the asset location
From Mile	The "from" mile where the work took place
To Mile	The "to" mile where the work took place
Wono	The work order number associated with the line
Cost Type	The cost classification
Item IDs	
Item Description	Description of the specific work order number item
Usage	The number of hours the resource was used
Total	The total cost for the item listed
Comments	This lists any additional user comments that were included in the IMMS entry.

One of the more complex fields to understand is the "IMMS Asset" field. An example value is "07-LA-110". This asset code helps to identify the location of a specific repair at a macroscopic level. First, the "07" is in reference to the specific Caltrans district. The "LA" is in reference to the specific county the work occurred in. The "110" is in reference to the highway for which the work occurred on.

Another field that is complicated to look at is the "Cost Type" field. This field is intended to classify the type of cost associated with the work order. This field has three standard values. These values are given in Table 5 below. Many times it is helpful to look at all items listed for a given work order number. Many work order numbers are listed without any reference to one or more of the asset types. This would indicate a work order which does not accurately reflect the entire cost of the repair. For example, a work order number which only contains material (M) with no entries in regards to crew (C) seems like an incomplete entry. This field was also integrated into the main CAL-COST tool when the user selects the "itemized" traffic control option in order to maintain synchronization of information.

Table 5 IMMS asset values and definitions.

IMMS Asset	Meaning
M	Material
E	Equipment
C	Crew

This section is meant to present a brief overview of the IMMS fields. The relevant terms are explained and the additional explanations are presented if required for explanation of the IMMS reducer. This is meant to serve as a starting point for the explanation to follow. Since the utility is focused primarily on the total cost of repair, it stands to reason, that all the information associated with a single work order number can be consolidated into a single piece of information.

Consolidating the Data

In order to consolidate the data, the work order number was used. The key information needed is the repair cost. This would be the sum of the "Total" costs associated with a given work order number. However, other key pieces of information are also relevant to the specific problem of this project.

First, many of the fields associated with specific items within the same work order number have common values. The utility gives the user quick access to these fields. These fields are as follows: Work order number, Comments, Date, IMMS Asset, To Mile, From Mile, District, and Activity. The utility that was developed simply presents this information as a single row entry as compared to the multi-row IMMS dataset.

Second, some quantities are derived values. One such derived value is the total cost associated with the work order number. The total cost associated with a work order number can be explained by equation (3) below,

$$\$_{Total} = \sum_{i=1}^{n} \$_i \tag{3}$$

where "n" represents the number of entries for a specific work order number, $\$_i$ represents the cost of the specific IMMS line item, and $\$_{Total}$ represents the total work order cost. The "$\$_i$" term is synonymous to the Total field in the IMMS database. Another key derived quantity is the average time or repair. This can be estimated in a few different ways. The way it is currently done in the utility has a hierarchy to the estimation process. The average time is computed in two different ways as shown by equations (4) and (5) given below,

$$\overline{Time_C} = \frac{\sum_{i=1}^{n_C} Usage_i}{n_C} \tag{4}$$

$$\overline{Time_E} = \frac{\sum_{i=1}^{n_E} Usage_i}{n_E} \tag{5}$$

where, n_C represents the number of "Crew" items, n_E represents the number of "Equipment" items, and $Usage_i$ represents the number of hours a specific item is used. This is the "Usage" field in the database. The utility then places a priority on the average time based on crew over the

average time based on equipment. The idea is that the labor cost will be more accurately entered than the equipment cost as it represents a higher percentage of the overall job cost. Also, it is believed that the labor cost will be more accurately sighted as it serves as a means to document employee performance and productivity. The equipment based average time is used when the average crew time is zero. The need for the two different formulations became necessary due to inconsistencies in what information is given to the database by the repair crews.

The last quantity which is somewhat derived is the material information. This is intended to give the user a quick sense to the completeness of the data associated with a work order number by indicating if the material cost used for the repair is included in the database. The idea is that many of the IMMS repairs do not include materials, and would make the cost of repair seem lower than it actually is. The caveat to this is that some attenuator repairs do indeed require no additional parts. This is more common in products which are considered “Severe Duty”.

Many other pieces of information are displayed by the utility. The idea is to summarize the information associated with a specific work order number into a single line. Many of the IMMS field values are consistent for a single work order number. The utility takes the uncommon fields and consolidates them to a single value which is a summary of the work order number. This process makes the information significantly less overwhelming and easier to decipher.

In order to understand the effectiveness of the consolidation process, it seems reasonable to look at an example. The first step is to get an excel source file from the IMMS database. Since the files can be extremely large, the example presented here will be attenuator repairs (Activity Code M80010) in district 4 from November 7, 2001 to April 14, 2011. The number of individual IMMS entries included in this file comes to 22,057 separate lines of raw data. The sheer volume of data is overwhelming and very difficult to manually sort through. However, by applying the consolidating technique mentioned above to reduce the information into a single entry for each work order number, the data can be reduced to 1,670 individual work orders. This process consolidates the data by 92.4%. However the issue of how to sort through the information still remains, albeit to a lesser degree. Various filtering techniques were developed which allow for the data to be quickly accessed according the specific task at hand.

Applying filters to the reduced data

After consolidating the data, the next logical step was to make tools to facilitate searching the data for very specific datasets. This is done by creating a range of filters that can quickly reduce the data in a way that is meaningful to the user. This section will briefly explain the filters used by the IMMS utility.

The mile filter utilizes the “to mile” and “from mile” fields of the IMMS database. However, since this information is not an exact science, it was felt that it was more important for the user to input a mean and a range instead of an exact number. The primary purpose to this filter is to aid the user in identifying work orders associated with a specific location that can be identified by post-mile. The filter eliminates entries which do not lie in the range defined by equation (6).

$$Mile_{mean} + Mile_{range} \geq IMMS\ mile \leq Mile_{mean} - Mile_{range} \quad (6)$$

The IMMS mile refers to either the "to mile" or the "from mile". Any work order number which has to and from miles that do not meet this requirement are filtered out. It should be noted that some IMMS entries only contain a valid number for the "from mile" or the "to mile" fields. Invalid entries would be those that do not exist, or have a zero value which seems out of place with respect to the other mile value. In these situations, the utility determines which mile field is non-zero and set the other field to that value. This corrects any holes in the data which would force valid data to be eliminated by the utility.

The district filter is intended to sort through the data with a specific district in mind. This filter eliminates all work orders outside of a specified district. In a case where the raw IMMS data is pre-filtered by district, the IMMS utility essentially disables this filter, since there is only one option available. This filter can be helpful as it makes exporting the IMMS data from the Caltrans database simpler. However, there is a size limitation to how big of an IMMS excel file the utility can process due to limitations in Matlab. By pre-filtering the data, the risk of reaching this limit is greatly reduced. Ultimately, this utility will need to be rewritten by a software developer, and a more robust approach can be taken.

The word exclusion filter allows the user to eliminate work orders by key words. This filter looks at the comments field associated with a work order number and eliminates any work-order with that word. This filter becomes more helpful when trying to identify specific products. One example of using this filter is to find repairs associated with a Smart Cushion. Clearly all references to "barrel" have nothing to do with a Smart Cushion. Therefore, any work-order that talks about barrels is irrelevant to what the user is looking for. Care should be taken to include words which are clearly specific to something which can be eliminated. Another example is to use this filter to look for impacts associated with a REACT350. If the keyword "barrel" is used, some REACT350 entries may be eliminated as some people may refer to the REACT cylinders as "barrels". A more appropriate word may be using the term "sand". The user can apply a list of words to this filter. For example a list of words such as "barrel,sand,REACT" (no spaces) would be a very good word exclusion filter when looking for Smart Cushion repairs. Please note, the omission of the word "and" in the list above as this will cause errors in the filter.

Another type of keyword filter is the word inclusion filter. This filter only keeps work order numbers which have direct references to the keyword. This filter is useful when trying to find all information that can be positively identified with a specific product. Where the exclusion filter is subject to user interpretation of the comments, the inclusion filter allows for entries to be identified which can be positively associated with a specific product though the IMMS data alone. An example of this is to use the word "Smart" for the search word. This will eliminate any work order that does not have a specific reference to "Smart".

The activity code filter allows the user to limit the information to a single activity. The filter will allow the user to select a specific activity code from a list of activity codes that are referred to in the IMMS source file. This filter requires the user to have some idea on the Caltrans activity codes. However, this will allow the user to greatly reduce a large data set which has not been pre-filtered based on activity code. However, it is still important to keep in mind that pre-

filtering the IMMS data before exporting to excel is the best way to ensure that file size limitations to not come into play as well as speeding up with filtering process.

The asset filter allows the user to only show data which is associated to a specific asset code. A sample activity code would be 04-ALA-880. The "04" is in reference to the district in which the work took place. The "ALA" is the county abbreviation which is Alameda. The "880" is in reference to the highway (or interstate) where the repair was done. The software implementation of this filter allows for multiple asset codes to be included. This helps identify repairs associated with a highway interchange, where the highway is included in the asset code is unclear or inconsistent. The asset filter requires special formatting in accordance with the IMMS database. The general format is "dd-CA-hhh" where "dd" is the 2 character district, "CA" is the county abbreviation (2 or 3 character in length), and "hhh" is the 3 character highway number. This filter aids in isolating data for specific highways. For example, highway 5 would be 005 and highway 101 would be 101.

The highway filter is for identifying repairs associated with a specific highway. This filter is somewhat redundant when compared to the asset filter described above. However, the input required by this filter is more intuitive as the user only has to enter the highway. In many ways, this is the cruder filter, as the asset file also filters county and district.

The county filter allows elimination of repairs that are not in the specified county. The user can select the county of interest from a list of counties which are included within the loaded data set. Reducing the counties by detecting them within the data helps to limit the user options in a way that makes the user interface more robust and intuitive.

There are some additional notes on some of the filters which need to be addressed. For the Word Ex. Filter, Word Inc. Filter, Asset Filter, and Highway Filter all user inputs must be separated by commas without a space. Efforts may be made to resolve this issue, and will be noted in future documentation. The word filters have some additional considerations which should be mentioned. Special attention should be paid to single or plural words. For example, in the word exclusion filter if the word "barrels" is used, all instances of the word "barrel" will not be excluded. However, the opposite is true if the word "barrel" is used then all instances of "barrels" will be eliminated. Complex phrases such as "sand barrel" will create errors. The user could instead try "sand-barrel" instead. A last note is that the word filters are not case sensitive.

The idea behind the filters is to allow for quick access to the data contained in the IMMS database and to facilitate mining the data for information in regards to a specific issue. There are two primary uses for this utility. One key use is to positively identify repairs associated with a specific product. Although these repairs may be incomplete, they help establish initial values for the repair metrics used by the CAL-COST tool. Another key use is to help identify the documented repair history at a specific site. This can be helpful in understanding the impact history of a given location, which can then serve as a basis for the impact characterization required by the CAL-COST tool. It should be noted that in both instances, some user interpretation is required as this is not an exact procedure.

IMMS Post-Processor Software

The IMMS Post-Processor utility is meant to serve as a mechanism to sort through the overwhelming amount of information contained in the IMMS database very quickly. The basic principle of the utility is explained above. The current form of the utility is a work in progress, and hence, there are some aspects of the utility which still need to be refined in order to be seamlessly integrated into CAL-COST utility. Other aspects of the utility could be considered scratch work and are primarily meant to demonstrate other possibilities for the tool. These aspects will be briefly discussed here for illustration.

Once the IMMS utility is accessed, a series of windows open up that are intended to relay information to the user. The first window which opens up is the warning dialog shown in Figure 33.

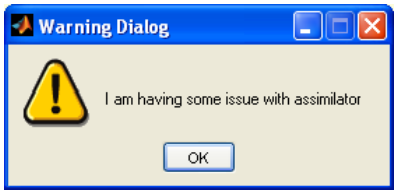

Figure 33 Program development reminder.

As this utility is still considered a work in progress, this warning is intended to be a reminder to the programmer of an unresolved issue in the utility. “Assimilator” refers to the portion of code which is being developed in order to integrate new IMMS entries into the CAL-COST resource file. This window is presented here to ease any concerns of the user as acknowledging known issues seems more appropriate. After pressing the “OK” button in the window shown above in Figure 33, another window pops up as shown in Figure 34 below.

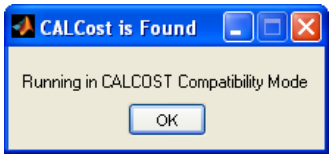

Figure 34 CAL-COST run-mode note.

This window has to do with the utilities development process as well. The IMMS utility can be executed separately from the CAL-COST tool. This dialog box notifies the user about what mode the utility is operating in. The typical user will see this window. However, some users will get a different message which indicates lower functionality of the utility. The user can simply press “OK” in this window and continue. The IMMS utility was written in a way that allows for utilizing the software for other applications that are not limited to only attenuators.

The first functional window of the utility is shown in Figure 35. This window presents a table to cross reference county and district. The list gives the county abbreviation followed by the full county name with the specific district listed in parentheses. The intent is to provide a reference

for those who are less familiar with the districts and county abbreviations. Some users may find this helpful, while others may simply press the "OK" button to move on.

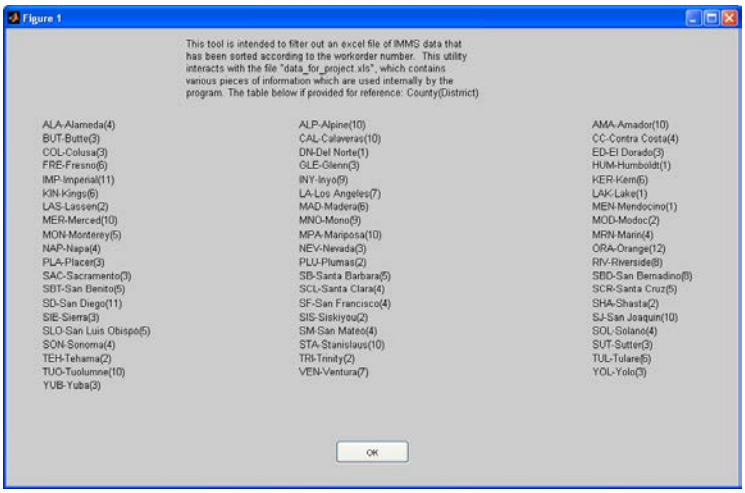

Figure 35 Program table showing county abbreviation, country, and district.

The next step in the utility is to load a data file. The utility will prompt the user to do so as shown in Figure 36 below. This file will serve as the starting point for the IMMS utility.

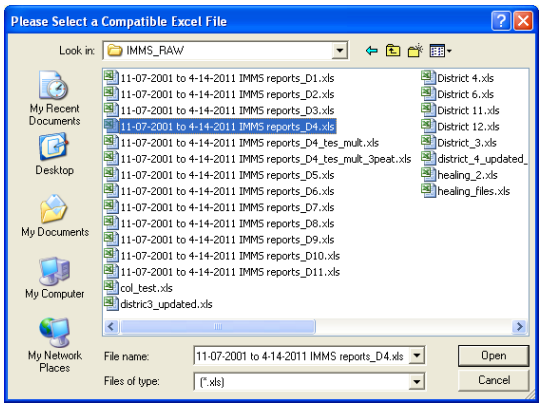

Figure 36 IMMS source data file selection screen.

The post-processor relies on the fact that an excel file generated from the IMMS database contains a somewhat standard format. However, it does have the ability to recover files which do not follow the standard format. The post-processor allows for remapping of the database column headings to fit the IMMS utilities pre-defined data structure. Once this is done, the utility can either proceed leaving the source file as is, or change the headings permanently in the IMMS source file to avoid this issue when the file is accessed at a later time. However, it should be noted that the second option permanently modifies the source file.

After a file is selected, the main IMMS window will appear as shown in Figure 37. The most significant portion of the window is the table in the center which contains a list of all the work order numbers that have not been filtered out. When the utility is first opened, the table contains

a single row of information for each individual work order number contained in the source file since no filters have been applied.

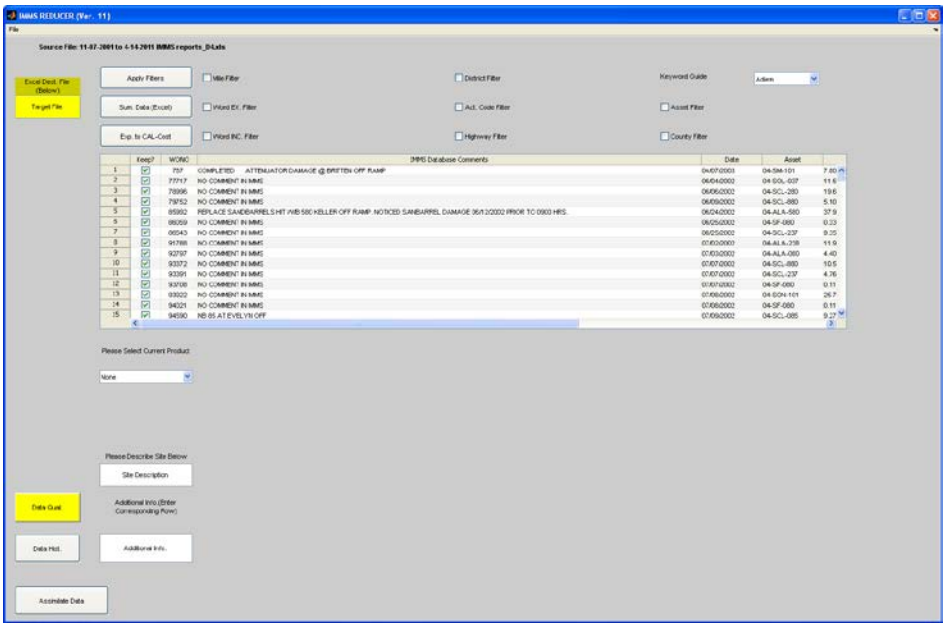

Figure 37 Main IMMS window at startup.

The horizontal scroll bar in this table can be used to view other fields in the IMMS data set. There are two additional fields which are included in this portion on the utility. Before a file is processed, the raw excel data is sorted according to the work order number. After that is done the "DB START IDX" and the "DB END IDX" are identified. These indices identify the row which corresponds for the first and last entry associated with a given work order. These indices are used when showing the detailed information about a given work order number.

In some instances the user may wish to look at the details of a specific work order number. This can be done by clicking on a table cell that lies in the same row as the work order number of interest or by entering in the specific row number in the text box under the text "Additional Info. (Enter Corresponding Row). Once this is done, a new table will be revealed as shown in Figure 38 below that contains all the raw IMMS information associated with a specific work order number. As the user can appreciate, the work-order selected above has quite a few individual IMMS entries which are not displayed.

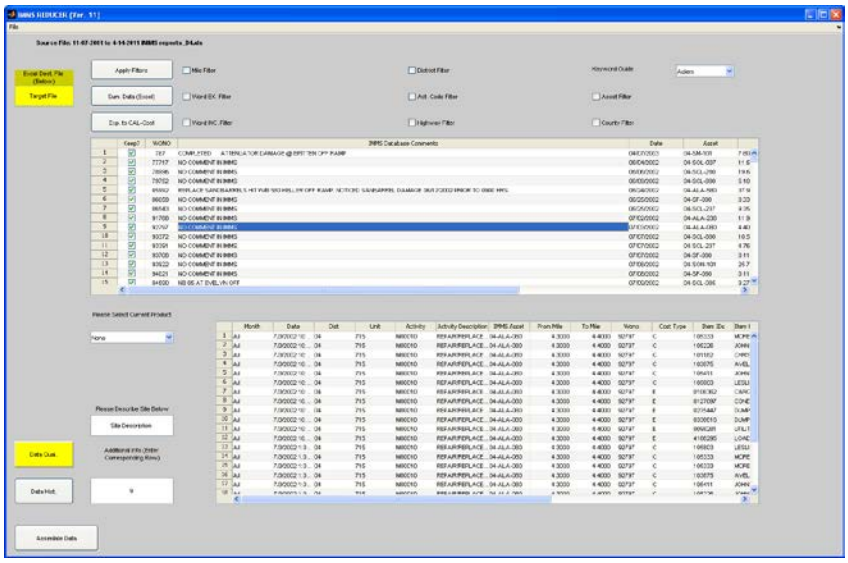

Figure 38 IMMS utility showing the work order details.

The next main area of the window is the filter section. This is the top section of the window that contains the series of checkboxes. These are the filters discussed in the section above. Figure 39 below, shows the utility with all the filter checkboxes selected revealing their associated input fields.

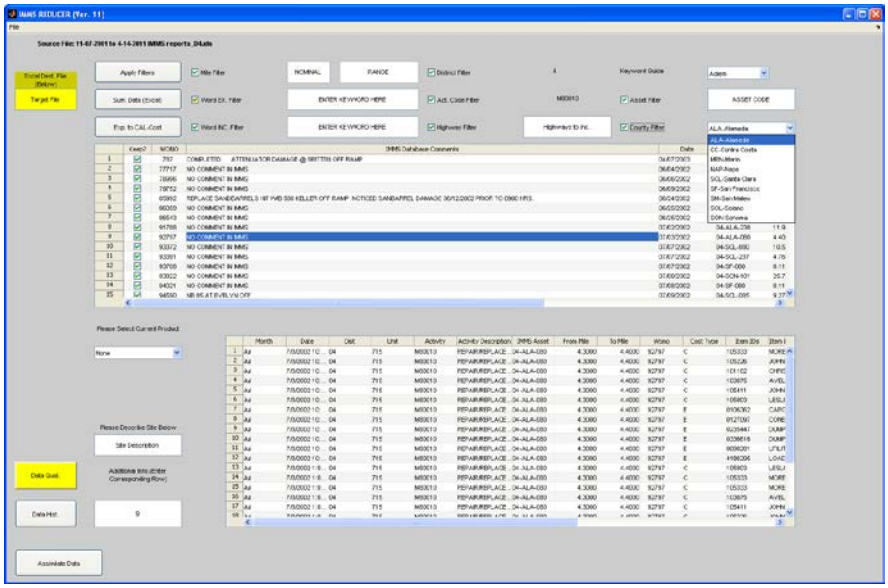

Figure 39 IMMS tool showing the county filter.

The most notable aspect of the window shown in Figure 39 is that the user input fields for the "District Filter" and the "Act. Code Filter" are pure text and do not give the user the ability to change the values. The reason for this is that the selected source file contains only one value for these fields. In the case where multiple values exist in the source file, these input values become pull-down menus containing all the possible choices contained in the source file. This is similar

to the "County Filter" as only counties tied to district 4 are shown. Table 6 below will give a summary of the filter and a brief description of the user input.

Table 6 Summary of filters implemented in the IMMS utility.

Filter	Input Format	Notes
Mile Filter	Nominal = number, Range = number	Nominal – Range > 0
Work Ex. Filter	Word1,Word2,…	Words should be separated by commas with no space
Word Inc. Filter	Word1,Word2,…	Words should be separated by commas with no space
District	Pulldown	
Act. Code Filter	Pulldown	
Highway Filter	Highway#1,Highway#2	Highways should be separated by commas with no space
Asset Filter	District-County-Highway (dd-c-hhh)	The county abbreviation may be 2 or 3 characters long and multiple entries should be separated by a comma only
County Filter	Pulldown	

Another key tool in the IMMS post-processor is the keyword guide which provides the user with a reference tool for sorting the data. Say the user wishes to look for information with emphasis on repairs associated with a Smart Cushion. The user can select "smart cushion" from the pull-down menu shown in Figure 40 below and get a series of lists as shown in Figure 41.

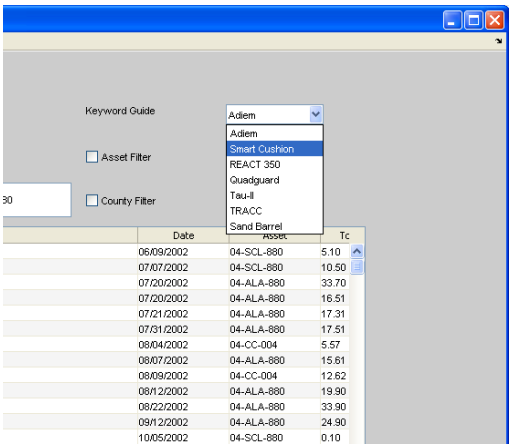

Figure 40 How to access the keyword guide.

The lists contain a series of buttons which will briefly describe the logic behind the specific recommended word. One example of this is if the user pushes the "Sand" button, the following window shown in Figure 42 appears. This part of the utility was intended to provide some guidance to words chosen for both the word inclusion and exclusion filters. This portion of the utility is under development and as the utility is further developed, the list may expand. However, the intent of the part of the utility is to help guide the user in regards to word choice.

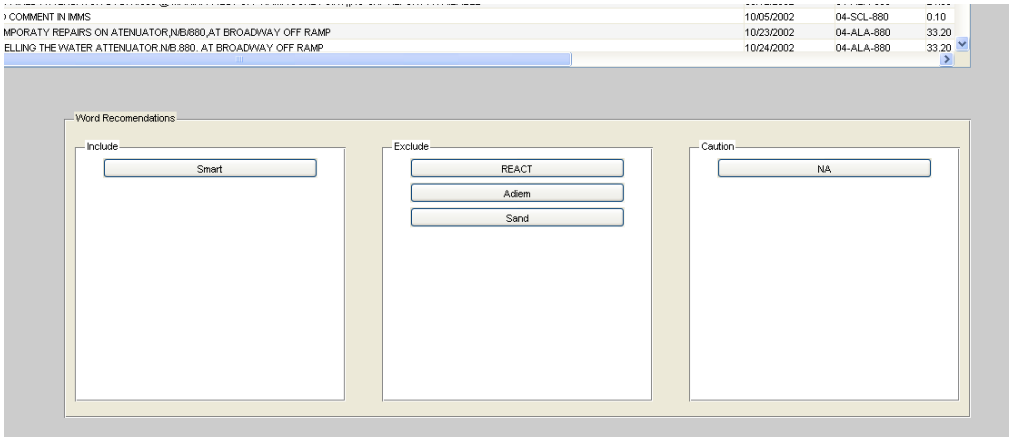

Figure 41 IMMS utility showing the keyword recommendation panel.

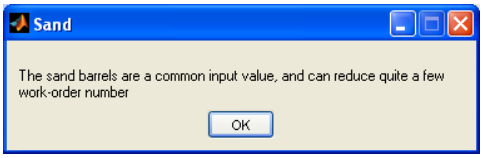

Figure 42 Example of a keyword recommendation.

After the user has selected the filters of interest and provided the relevant inputs, the user simply needs to press the "Apply Filters" button to apply them to the data. The filters can be repeatedly applied and modified to aid in honing in on specific pieces of information.

Another useful utility is the word search utility. This can be accessed from the file menu, pressing "Ctrl+F", as well as by right clicking in the main data table. The utility allows the user to test a word for use in either the inclusion or exclusion filter. Once this is activated, the window shown in Figure 43 appears.

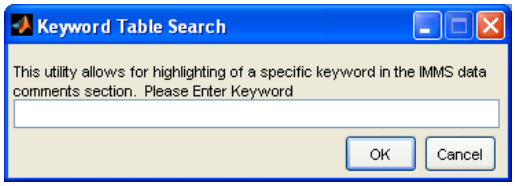

Figure 43 Keyword search input box.

Here the user can type in a specific word, say "barrel" for example. The utility goes through the comments field in the table and highlights all instances of the word as shown below in Figure 44. The idea was to give a preview of how many instances a word appears without actually filtering the data. In essence, it will provide a much quicker over-view of the effectiveness of a specific word filter.

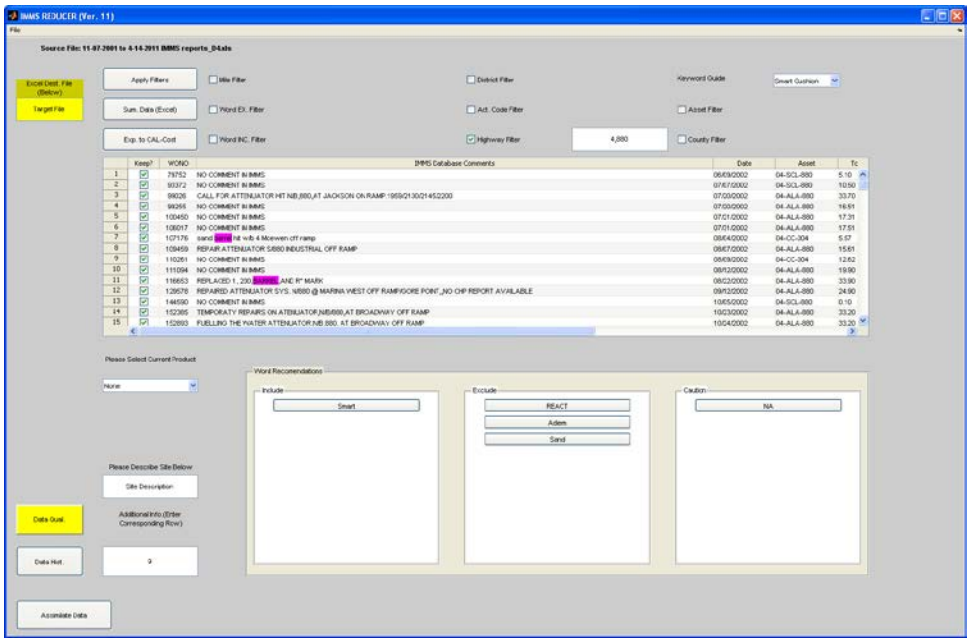

Figure 44 IMMS showing a keyword search.

After the data is filtered as desired, it can then be exported to the CAL-COST tool. Any specific work order numbers that should not be exported for various reasons, they can be simply unmarked using the checkbox in the "keep?" column of the table. The "Exp. to CAL-COST" pushbutton will transport the findings into the current site window of the CAL-COST tool as shown in Figure 117 (if the system in running in the CAL-COST compatibility mode). A pull-down menu is available for the current product to be entered in as well in the IMMS post-processor. However, this can also be assigned in the current site window of the associated CAL-COST panel as well. A text box where a site description can be entered is also provided. Both the specified product and site description will transfer over to the current site window illustrated by Figure 117. The information associated with these fields is not required. The main benefit of this process is to allow for direct reference to the sites specific work history to be integrated into the CAL-COST analysis in a streamlined fashion.

Another way to save the findings is to save the summarized information in an Excel file. This was originally developed as a means to quickly save search results and look at them later. For example, if there is an alternative use of the IMMS utility, independent of the CAL-COST project, this file could be used as a means to export the desired information. However, if this utility becomes more widely used, this feature may be eliminated or moved depending on user experience and feedback.

IMMS Post-Processor Development

The section above outlined the core functionality of the IMMS post-processor. However, there are some additional options which are not completely refined and are still in the development process. There are two key areas that need further refining in the entire CAL-COST process. The functionality presented here is merely done for demonstration purposes. As the project moves

forward, and more information is incorporated into the utility, efforts will be made to refine this information and help establish a more robust interface.

One area which needs to be looked at is a means by which to measure the quality of the data. Currently most of the obtained data is derived directly from IMMS and not corrected for things such as access cost, lack of material/parts cost, etcetera. Therefore, the data is not high quality; however, it does serve as a starting point of information. There is a button in the IMMS tools screen labeled “Data Qual.”. Figure 45, below is meant to serve as a glimpse to the intent of this button. As one can see, the current data is either preliminary or raw IMMS data. Efforts will be made to further refine this as deemed necessary.

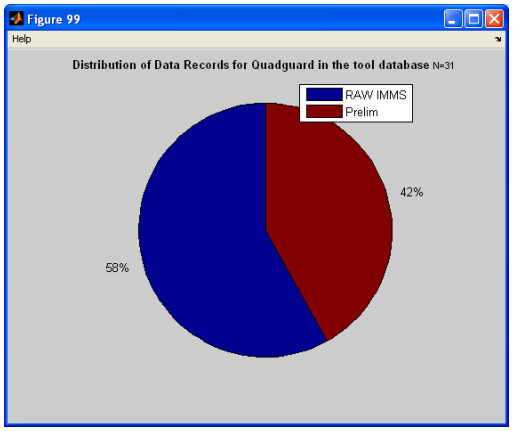

Figure 45 Data category distribution.

Another plot which is associated with data quality is the data histogram. The plots shown in Figure 46 are purely representative. However, the intent is to illustrate the quality of the product averages as well as to aid in identifying data which appear to be outliers. Currently these plots have a fundamental flaw which may account for the data not following the intuitive Gaussian trend which is attributed to that fact that access cost is not properly accounted for in the raw and preliminary data. If the data does not have the correct quality to it, these plots will be less likely to follow the expected trends.

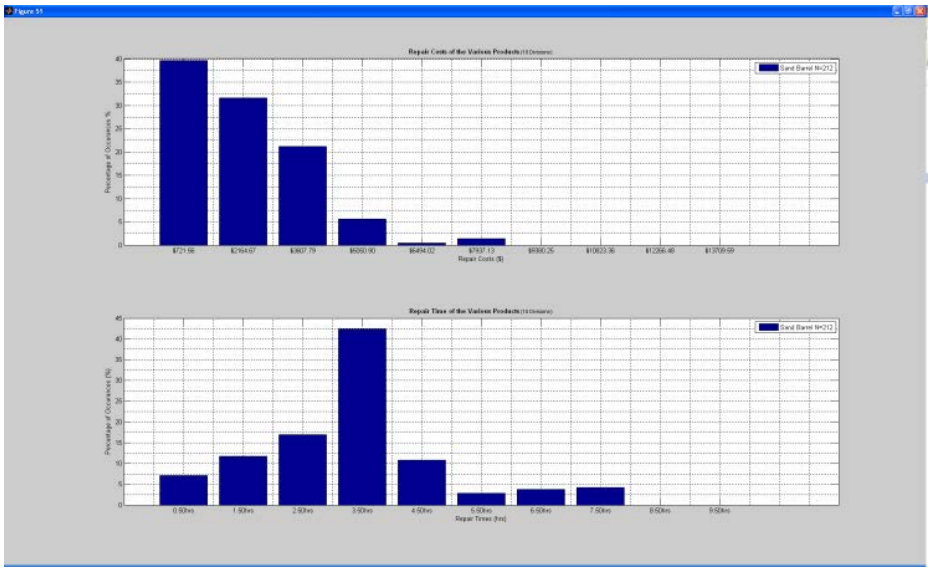

Figure 46 Data quality histogram.

Although there are many flaws with the information above in terms of what information is captured by the data, this is meant to illustrate that efforts need to be made to assess how good the information is before any detailed conclusions can be made. The goal is to work with Caltrans to utilize the CAL-COST process and in doing so refine the data that is used so that the data quality becomes more robust.

Another extra feature is that the system allows the findings of the IMMS sorter to be integrated into the IMMS resource file. This will serve as a starting point for the data. Then the data can be specifically modified by contacting people within a DOT organization in order to refine the numbers. One thing that should be noted with the resource file assimilation is that extra care needs to be made to prevent duplicating information. This "check" is very important, as it prevents a single repair from biasing the data; however, it does come with a huge cost in computational time in its current implementation. Another major drawback to this process is that there is no definitive piece of information within the IMMS database that ties the repair to a specific product. This means that it is up to the user to be very rigorous in identifying the product associated with a specific work order number.

Summing it up

The section above was meant to illustrate the IMMS reducing utility. The key intent of this utility is clearly demonstrated through the implementation presented here. Generally speaking, the intent is to give the user a mechanism to quickly sort through raw IMMS information in order to get a basic understanding of a specific site. An additional use for this process is to understand work order costs associated with a specific product. This information can inform the designers on what is going on at a specific site. After understanding the IMMS specifics, any additional questions and or concerns can be answered by contacting the specific maintenance crew about the location.

Appendix B

Camera Integrated Impact Sensing and Monitoring System Design

This appendix provides the detailed design and testing of the impact sensing and monitoring system developed in this research project. Since there are many camera monitoring systems commercially available, a suitable system was selected and was combined with other sub-systems to have fully integrated impact sensing and monitoring system.

Camera Selection

After identifying the key functional requirements of the camera system, a search was performed to identify possible solutions. The Logitech Alert 750, shown in Figure 47, seemed to meet the design requirements. The camera is capable of recording at night by using infrared (IR). The IR mode is only activated when the camera's internal software determines it is appropriate. This is determined based on lighting conditions. The user has little control on switching between these modes, but the camera's brightness and contrast settings provide some control over the switch mechanism. Since infrared light is not visible to the human eye, illuminating the site using an IR illuminator will not significantly affect the site's properties. Additionally, Logitech provides an interface which allows a user to set up motion zones within the camera's field of view to define an area of interest for motion detection. The camera's built in image processing software looks at these zones in order to detect motion. The zone definition will help to eliminate false triggers by allowing distinction between cars that are just driving by and those that have impacted the attenuator. There are various parameters and motion zones associated with a triggering event can be configured through the internet. This allows the sensor to be tuned in the field. Once the camera is triggered, a small video clip surrounding the specific event is locally stored. The video data can be remotely viewed and downloaded through a web interface. Additionally, the Logitech software can be configured to send immediate email notification to the designated address when a triggering condition is met. This camera is intended to operate through a network connection on either a home network, or via the web utilizing a POE interface.

Figure 47: Logitech camera (www.logitech.com).

Camera Test

Since the Logitech 750e camera was designed for home security use, it was necessary to test its capabilities in a simulated highway environment. One important characteristic of the environment is the lighting condition that changes throughout the day. The main objectives of the tests were to determine the camera's displacement resolution and motion-detection capabilities.

Determining the displacement resolution was necessary because self-restoring crash attenuators, such as the React 350, can experience nuisance hits and have no discernible damage. The minimum amount of displacement detectable by the camera would set the displacement threshold for a nuisance hit. It was expected that this displacement resolution would be inversely proportional to the distance from the camera. Additionally, the displacement resolution would be lower when the camera was in night mode due to the lower resolution images and the reduced frame rate.

The motion-detection capabilities of the camera were tested for movement of the crash attenuator itself, movement by vehicles, and sensitivity to passing vehicles. It was expected that movement of the crash attenuator would trigger the motion-detection of the camera. However, since some self-restoring attenuators experience minimal movement in a nuisance hit, it may be possible for a nuisance hit to avoid triggering motion-detection. Such would be the case on a glancing impact on Smart Cushion. The structure is so rigid that very little lateral displacement would be perceived by the camera. If this was an issue, then it was hoped that extending the motion zone around the crash attenuator would enable the movement of an impacting vehicle to trigger the system due to it being in close proximity to the system. Finally, there were concerns that passing vehicles would produce false motion-detection. The fact that this system can be tuned via the web would allow for the systems to be tuned for specific sites once they were deployed.

Test Setup, Procedure, and Results

Initial testing was done on the ATIRC test track. Black-painted barrels were used to simulate an attenuator as shown in Figure 48. The tests were carried out under three lighting conditions: day, dusk, and night. The nighttime tests were done with and without active IR lighting to assist the camera's night mode. The farthest barrel of the setup was placed from 30-60 feet from the camera. Finally, a midsize pickup and a midsized sedan were used as vehicles for the testing.

Figure 48: Night mode with active IR lighting.

Displacement resolution (Table 7) was measured by placing down a tape measure at each distance and noting the minimum amount of movement visible through the camera display. It

was important to determine the displacement resolution for the camera's two modes: day and night.

Table 7: Displacement resolution.

Mode	Distance			
	30 ft.	40 ft.	50 ft.	60 ft.
Day	1 in.	1 in.	1 in.	2 in.
Night	1.5 in.	1.5 in.	1.5 in.	2 in.

The motion-detection was measured through three tests. Attenuator motion was tested by displacing the attenuator at several speeds and noting any resulting motion-detection. Vehicle movement was tested by extending the motion-zone slightly beyond the attenuator and passing the vehicle through the motion-zone. Finally, passing vehicle sensitivity was tested by passing the vehicle outside the motion-zone.

Slow movement of the attenuator was unable to activate motion-detection in various lighting conditions and distances. Fast and medium movement of the attenuator was able to activate motion-detection in all lighting conditions and distances up to 60 feet. Based on this testing, it was anticipated that the movement of the attenuator from typical impacts would activate motion-detection, but it should not be relied upon if slow impacts or minimal movement of the attenuators are expected.

Vehicle motion was detectable at 30-50 feet under all lighting conditions. At 60 feet, roughly one-third of vehicle motion was detectable. Extending the motion zones would be a useful supplement to detect impacts in lower speed situations and for distances up to 50 feet.

There was varied sensitivity to passing vehicles. During the day, there were false detections from vehicle shadows at the closer distances up to 30 feet. Motion zones should be carefully drawn to minimize effects from shadows. Similarly, at dusk there were false detections from vehicle headlights up to 30 feet. The camera's brightness should be turned down to minimize headlight effects. In night mode without active IR lighting, there were false detections from vehicle headlights at all distances. In night mode with active IR lighting, the false detections were present at distances past 40 feet, where the active lighting had less intensity.

In summary, the vendor supplied motion detection showed favorable results for this project. Initial testing suggested that active IR lighting is required and the camera brightness should be turned down to minimize headlight effects. A mechanism to assist in segregating out false triggers from events would make the system more user-friendly. An active motion zone trigger seemed to be a possible solution. In the sites that were selected, the ambient light was sufficient for nighttime monitoring that the IR illuminator could be disconnected. However it seems appropriate to keep this as a viable option in order to maintain flexibility for various installation conditions. Therefore the implementation of the IR illuminator will be included in the discussion to follow.

Camera Software

One of the design drivers behind selecting this particular camera was that the camera's manufacturer provided different software applications which eliminated the need to develop any image processing software. This streamlined the sensor development timeline as no resources were needed for major software development. Figure 49 below shows a general over-view of the camera's web interface. This enabled the research team to easily monitor the sites from any computer as needed.

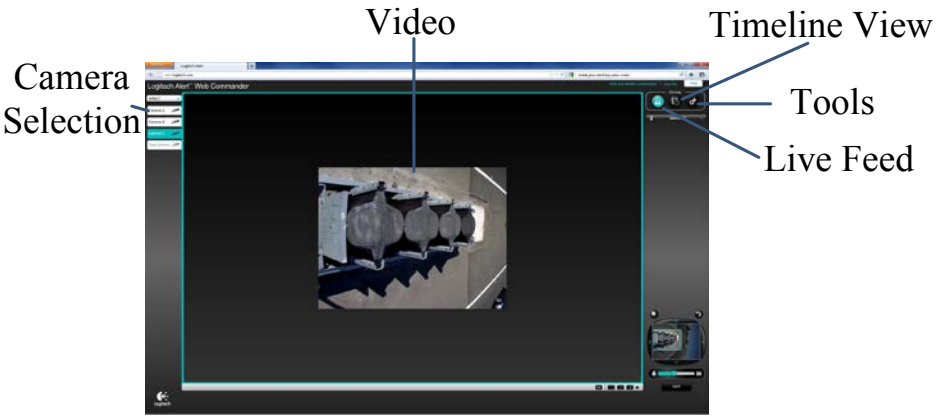

Figure 49: Live camera view.

The "Timeline View" button allows the user to switch the window to the time line view, allowing the user to see any triggered events stored in the camera's memory. This is indicated by vertical lines along the timeline as shown in Figure 50 below. By double-clicking on any event, the software plays the selected video clip. This clip then can be stored or shared based on what the user prefers by using the vendor supplied interface.

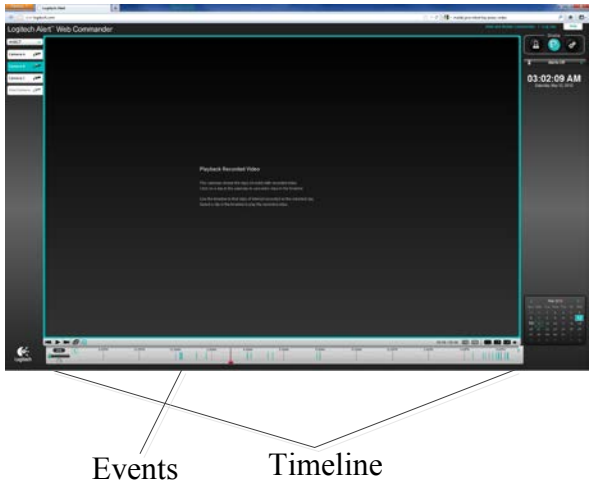

Figure 50: Timeline View.

The web application allows the user to adjust the camera's settings. The largest of which is defining the motion zones within the camera field of view as can be seen in Figure 51 below. The key to defining the motion zones is to keep the zones as small as possible to minimize false

triggers. The biggest issue here is the abrupt changes in lighting associated with the shadows that are changed by the headlights of moving cars, which is processed as motion by the camera's firmware.

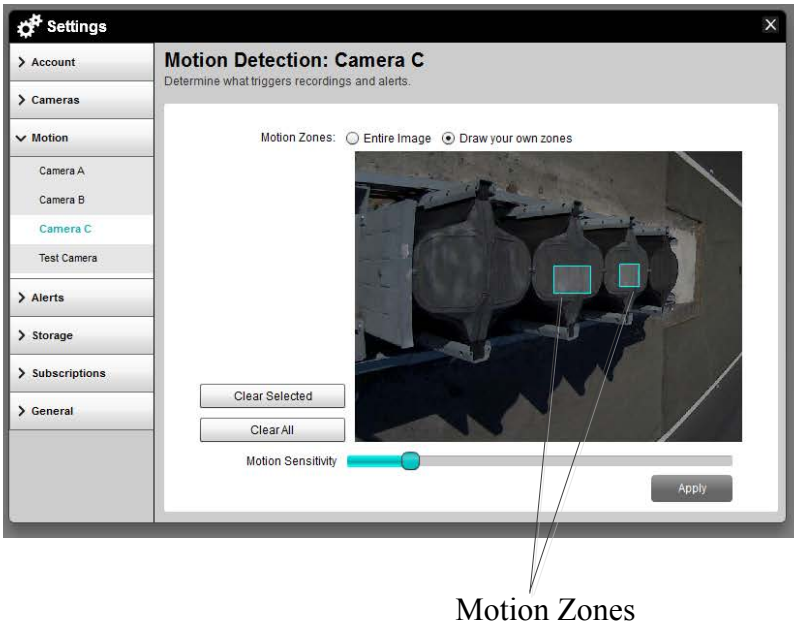

Figure 51: Motion zone example.

The motion zones were set by areas which were saturated by the IR illuminator at night. The idea was to create a washout effect in those zones so that any additional lighting provided by the traffic headlights would not create false triggers. During testing, it was shown that an impact will cause enough physical movement within these zones to trigger the system. Generally speaking, the motion zones will have to be adjusted on a case by case basis.

In addition to the software which is available for a computer, Logitech developed a smartphone application for both the Android and iPhone platforms. This application provided a mechanism to quickly verify system operation in the field. During installation, this application was used to "aim" the camera to get the best field of view to capture data by using the live feed option.

In general, the two main software interfaces provided by the vendor streamlined both the development and the deployment of the system. The web interface allowed the system to be easily reconfigured once deployed in order to solve any site specific issues. The phone application facilitated initial setup by allowing immediate access to the camera's FOV. The only major system requirement for this system was to provide the camera with a reliable network connection using a POE interface.

Raven Modem

Additional hardware is required in order to get the video off of the camera's internal memory. The camera does not need to be connected to a computer and can function directly with the web as part of the vendor supplied interface. A cellular modem is sufficient. For this system, the use of a Raven cellular modem seemed to be the best choice. This modem is essentially stand alone

and does not require a computer to operate. The modem runs off of a 12VDC power source with minimal current requirements. The single network port coming out of the Raven can easily interface with the camera. However, the Raven network port does not support the POE interface that is required by the camera. The 48VDC required by the camera can easily be spliced into the same network cable in order to power the camera. On the same port as the input power to the Raven, there are two IO pins as shown in Figure 52 below. The IO pins are high by default and change state when connected to ground.

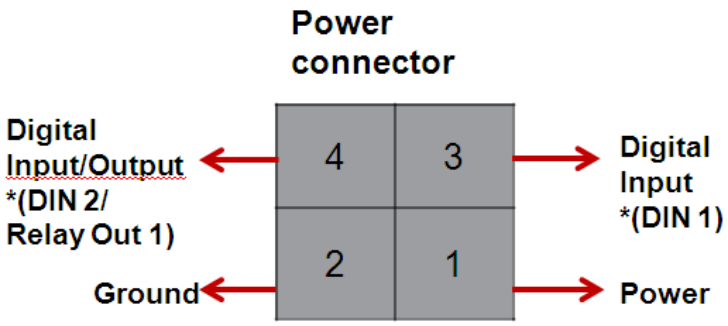

Figure 52: Pinout of Raven Modem.

The modem's ALEOS firmware can be accessed remotely using a fixed IP address. This can be accessed through a web browser using the web address of IP:9191 and logging into the system with a username and password. The software allows for events to be reported when the IO is triggered as the user sees fit as shown in Figure 53. Each modem is assigned a fixed IP address in order to simplify remote access.

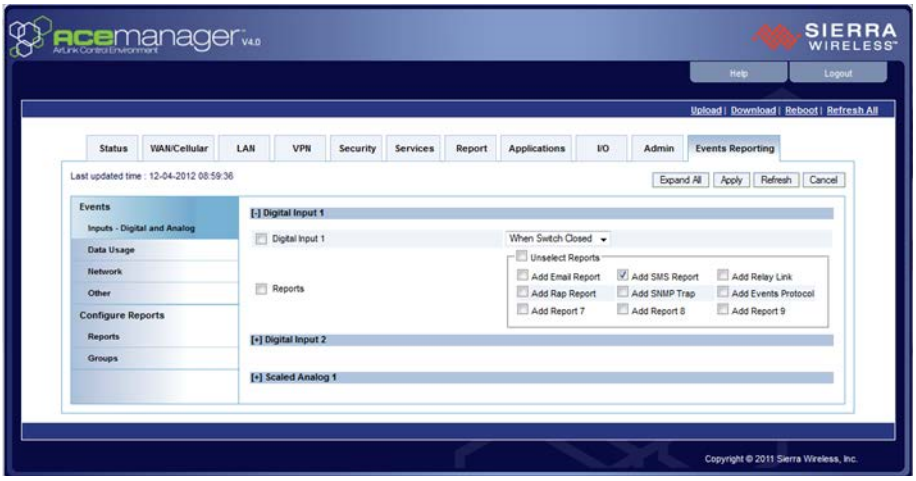

Figure 53: ALEOS Events reporting window.

The other aspect that needed to be determined is how the events are reported. The ALEOS software allows for two communication protocols. One is through the use of email, and the other is sending a text message to a cell phone using SMS messaging. The text messaging protocol was the most straightforward to configure for this system. This text message can be sent to a number that is associated with a VoIP site in order to keep a record on the cloud. This allows for

data to be accessed by each team member. During initial deployment, the message can be sent to a typical cell phone in order to check system functionality. Text messages are time stamped by default and can therefore serve as a reference point for the Logitech camera's timeline. This whole notification process is triggered by switching the specified IO pin on the 4 pin connector shown in Figure 52 above.

Infra-red Illuminator

Once the camera was purchased, testing was performed at the AHMCT test track to develop an understanding of the camera's IR capabilities as mentioned above. Built into the camera was a small IR illuminator. Testing showed that the built in illuminator had insufficient power for our application and an external illuminator was required. After looking at various options, a 60 watts IR emitter light bar seemed to be the best option (Magnalight LEDLB-20-IRS-850nmWHT). The light runs off of 12VDC which is compatible with the rest of the system's power. Since this would only be required at night, a simple photo sensor (CellOptik 12VDC "On at night") can be used in order to turn the illuminator on and off based on the sites ambient lighting conditions.

Microcontroller System

The microcontroller system was developed in a way to augment the existing camera system motion detection system. This has two effects. One effect is to provide a secondary time stamp to help sort out through any false triggers that utilizes the Raven's messaging protocol. The second is to serve as an additional mechanism for motion triggering, by creating motion in the camera's FOV. Physically, the system needed to be compact in order to minimize the enclosure size. The microcontroller utilizes an accelerometer to detect an event and interfaces with the two input pins on the Raven modem as depicted in Figure 52. This part of the system required extensive hardware and software development in order to create a reliable robust system.

Accelerometer Hardware and Testing

After identifying how to interface the microcontroller with the Raven, additional sensor hardware had to be identified. Implementing an accelerometer seemed to be the best option. One fundamental issue here is that typical accelerometers are analog signals. This inherently makes them a poor choice to interface with the closed contact inputs of the Raven XE. However, one advantage of using an analog accelerometer is that the signal can be processed in a way to establish a sensitivity threshold. This is done by placing a dead band on the sensor as the accelerometer can measure both positive and negative accelerations. This threshold allows for filtering out any false measurements due to wind generated by passing vehicles or roadway vibrations (more of a concern on bridge decks) that would be manifested as low level accelerations. The threshold is illustrated by Figure 54 below. Future versions of this system could utilize the full analog nature of the sensors to provide more complex information. However, this would require either a different modem system or a more complex microcontroller in order to develop a means by which to relay the more complex information to the user through a network interface.

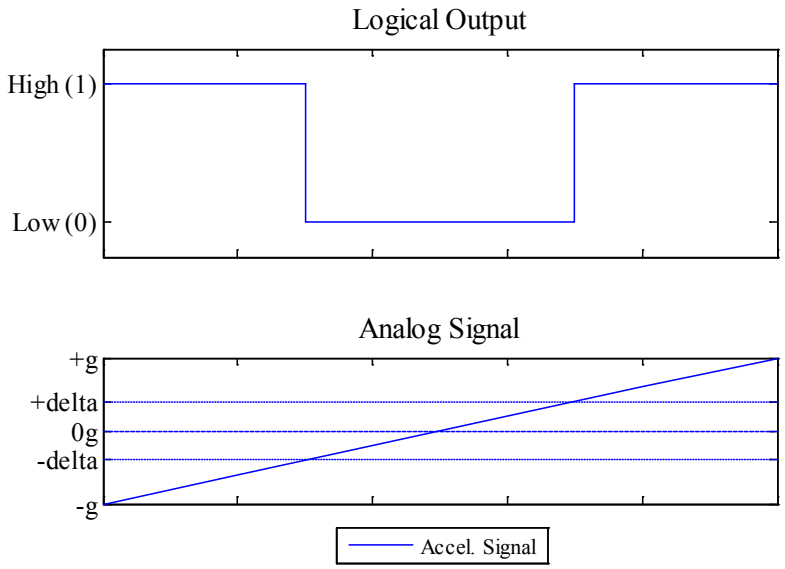

Figure 54: Sensitivity threshold.

Based on the information above, it is clear that some sort of processor will be required in order to take the analog signal and convert it to the logical output. The goal is to have the logical output serve as the notification signal used to trigger the Raven's event reporting routine. One simple microcontroller is the Arduino Uno (Pololu 2191). This is a low cost control board that is commonly used by hobbyists Figure 55. The cost of the board is approximately $30 and it is capable of running off of a 12VDC bus. The standard board has 6 analog IO's and 14 digital IOs. This has sufficient intelligence to meet the system's needs. Another huge advantage to this product is the extensive user community which facilitates rapid development of the micro-code and any additional hardware development.

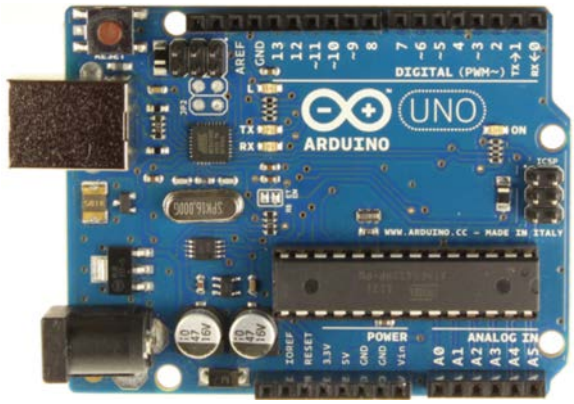

Figure 55: Arduino UNO (www.http://arduino.cc/en/Main/ArduinoBoardUno).

After identifying a candidate for a controller, an accelerometer was identified. Pololu robotics carries a 3-axis accelerometer that operates in two different range modes: ±3 or ±11 (Pololu item #: 1252 MMA7341L 3-Axis Accelerometer ±3/11g with Voltage Regulator) which is shown in Figure 56 below. The range is selected by applying 3.3VDC to the G-sel pin of the

accelerometer. These modules are commonly integrated with an Arduino controller which facilitated implementation into this system.

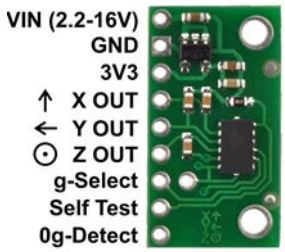

Figure 56: Accelerometer http://www.pololu.com/catalog/product/1252.

For implementation, the software is configured so that the 0g baseline reference value for each axis is defined as the nominal sensor value at rest. The value is defined when the microcontroller initially powers up and shortly after a triggering event. The intent is to prevent the system from reporting impacts when the system's baseline has shifted. This issue also raises the concern that the system needs a mechanism to prevent it from continuously sending messages. This is mitigated by putting in place a system which limits the number of messages that can be sent over a certain amount of time and part of a health monitoring system that is summarized below.

Physically connecting the accelerometer to the Arduino is trivial. The power connection (VIN and GND) can either be wired to the Arduino power circuit, or externally. The X OUT, Y OUT, and Z OUT pins of the accelerometer are connected to three of the analog channels of the Arduino controller. The g-Select pin sets the range of the accelerometer from ±3 or ±11. This is done by applying 3.3 volts to this pin which is available from directly from the Arduino. The Arduino utilizes a 10 bit analog to digital converter. This means that based on which g range is selected, the sensitivity resolution is affected. However, since the analog sensor is being processed into a binary message, the resolution is not a major factor. In the deployed system, it was decided to put this on a switch to give added flexibility in configuring the system when deployed in the field.

A simple system was built in order to test the accelerometer triggering system in the field. The goal of testing was to determine the threshold for the accelerometer filter in order to have sufficient sensitivity to capture small impacts, while minimizing any false triggers due to vibration. The accelerometer was placed on the rear support structure of the attenuator at a location that would not affect the functionality of the system. The system was then monitored for a period of time in order to see if any false triggers occurred due to any vibration induced into the system by passing cars, trucks, and semi-trailers. The threshold value was adjusted until no false triggers occurred. The next part of the test was to give the attenuator an input which would represent a low level impact. The input consisted of kicking the attenuator at a point that was furthest from the sensor in order to maximize any inherent vibration filtering effects caused by the physical system. The test system was triggered as desired by this test which clearly demonstrated that the accelerometer would be a very helpful addition to the sensor system. The threshold setting would require adjustment in the field in order to have the same level of functional performance for an assortment of products as each product would have different vibration propagation characteristics.

Implementing the Accelerometer System

Now that the two components are identified and tested, the electronics to get the Arduino to trigger the Raven's IO ports needed to be determined. This is done through a simple transistor circuit shown in Figure 57. The Raven IO pins are designed such that connecting the Raven IO pin to ground, creates logic low state on the IO pin. The functionality is such that by turning the Arduino pin to "HIGH", connects the Raven IO pin to ground and hence sets the value to 0. This was required as the Arduino IOs are not compatible with the Raven's IOs which are at 12VDC.

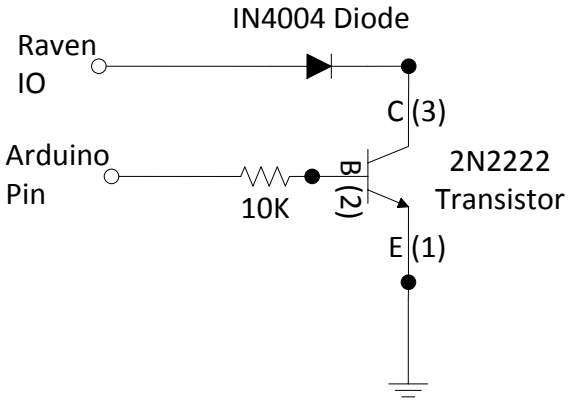

Figure 57: Basic transistor circuit to trigger raven IO.

The above circuit is sufficient to trigger the Raven IO; however, it was felt that it would also be helpful for development and troubleshooting to augment this circuit with an LED indicator. This modification is shown in Figure 58 below. These LED's allow for development of the Arduino firmware without requiring the Raven modem to be physically connected to the system. There are two such circuits integrated into the final design which correspond to the two LEDs. These LEDs are integrated into the microcontroller housing. In essence, these pins can be thought of as the "Flag" pin and the "Send" pin.

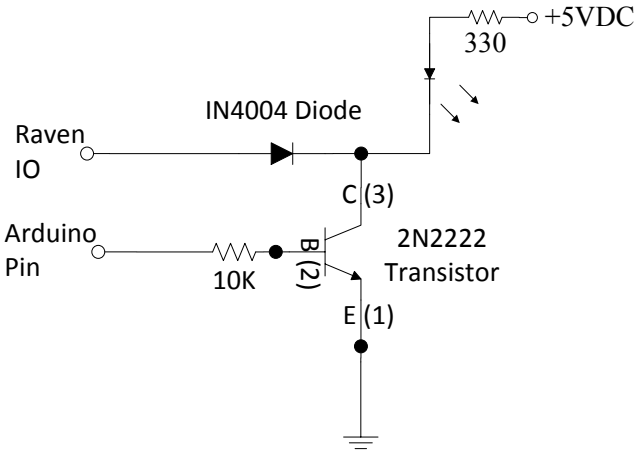

Figure 58: Raven IO circuit with LED.

The system sends three messages per impact, corresponding to the sensors x, y and z accelerometer axes. This will be done by setting the state of the flag pin, which will indicate whether or not that particular axis was actuated, then using the "Send" IO to trigger the event reporting system that is built into the ALEOS firmware.

The core basis of the accelerometer sensing system has been explained. However, by utilizing a microcontroller some additional enhancements were made to make the system more robust. The microcode on the Arduino looks at all three accelerometer values to determine if any of the three axes have a high logical output. Once a sensor is tripped, the microcode continues to monitor each of the other sensors for a short amount of time to see if any other sensor is tripped before going into an event notification mode. The logic behind this is that it will be difficult to capture all sensor values instantly due to the procedural nature of the microcode. After the short monitoring period is complete, the microcode begins to relay the output to the Raven. For a single trigger of the sensor, three messages will be sent in quick succession, corresponding to the x, y, and z axis accelerometers. The user can then look at the state of the flag pin to determine what happened to the sensor. An example of this is if the user got a group of messages and the digital input 2 values are 1, 0, and 0. This would mean that the x axis sensor was not tripped, but both the y and z axes were. The reader should also note that x, y and z axes are based on the sensor orientation. For this situation, the orientation is non-critical as the corresponding video will allow for a more detailed understanding of the event specifics. However, if the system's data communication scheme is modified to directly relay the accelerometer's analog measurement, the sensor orientation will be critical.

Dual Accelerometers

During system development, connecting two accelerometers to the Arduino made sense. Since the Arduino has six analog inputs, both accelerometers could be hooked up to the Arduino electrically. The Arduino code only actively monitors 3 analog inputs at any given time in the micro-code as monitoring 6 accelerometer channels continuously seemed excessive. However, in the event that the system detects an issue with any single accelerometer channel, the micro-controller will switch which accelerometer bank of sensors that is being monitored. An example of this would be if a conductor failed to one analog channel. This would induce an unreasonably high frequency of detected impacts due to a condition commonly referred to as a floating IO. This was added in order to make the system more robust and help reduce the need for servicing the systems in the field as the microcontroller has the ability the self-diagnose this issue.

Auxiliary Triggering

After developing the circuitry to trigger the Raven, a mechanism that allows the micro-controller to trigger the camera needed to be developed. The idea is to ensure that the camera has a recording near the time of messaging if the camera's built in software fails to do so. Since the camera is triggered based on motion, the system was configured in a way to flash an LED within the camera's field of view that is included in a motion zone. This will ensure that a video is captured when an event occurs. Through experimentation, it was found that blinking this LED with a 160ms pulse train with a 50% duty cycle was optimal for triggering the camera under the varying video capture settings. The LED initially served as a backup to the vision based triggering the system. However, as more experience was gained with the system, the camera

settings were adjusted to use this LED as the only motion detection trigger associated with impact monitoring.

Additional Features of the Microcontroller System

Since a microcontroller was used, there were some additional features which could be added to enhance the overall system functionality and usability. These features help make the system more robust and flexible.

Uploading New Firmware

Since this is a work in progress, identifying how to perform any software updates in a safe manner was important. This was done through the utilization of a pair of Wixel boards that are shown in Figure 59 below. One is connected directly to a computer and the other to the Arduino. These boards operate on a 2.4ghz radio frequency and have a range of about 30ft. There is a wireless serial application for the Wixel available that is written to specifically allow for uploading Arduino micro-code. The Wixel connects to the Arduino through a product called the Wixel shield available through Pololu robotics, shown in Figure 60. This connects the Wixel to the Arduino in a way to complete the integration of these two products in a single stack.

Figure 59: Wixel board.

.

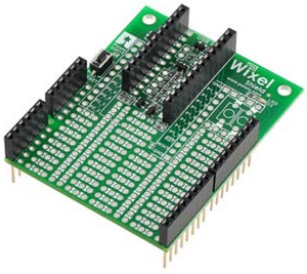

Figure 60: Wixel shield.

One benefit to this shield is that part of the board provides sufficient space to integrate additional circuitry into the system. There is sufficient space to integrate the circuits presented in Figure 57 and Figure 58 into the microcontroller stack. This allows all the additional hardware that is required to integrate the microcontroller with the other system components to be packaged in a very concise footprint.

General Sensor Operating Information

In order to give more options for the system, a pushbutton switch was added as a digital input to the Arduino. The basic circuit of this is shown below in Figure 61. This IO will allow for additional flexibility in the software. In the current version of the software, the button has a dual

purpose. In the event that the user presses and releases the button, the hardware will report back the key settings within the Arduino firmware which will be discussed later. The difficulty of this is that the user must establish serial communication without toggling the data terminal ready flag of the serial communication protocol. Toggling this flag will restart the firmware on the Arduino, and hence lose any values that have changed over time as the system runs. This is an inherent property of the Arduino UNO. The other functionality of this is that by pressing and holding this button, the system will completely reset itself. This is verified locally by watching the flag LED blink while holding the button. This is basically the same functionally as cycling the main power bus or toggling the data terminal ready flag when wireless communication is initiated. However, in utilizing this button, the power on the Raven and the camera is not cycled. This eliminated the need to wait for everything to completely reboot in order to test the system quickly in the field. The goal was to give as many options in order to troubleshoot the system in the field while allowing for any future un-anticipated enhancements.

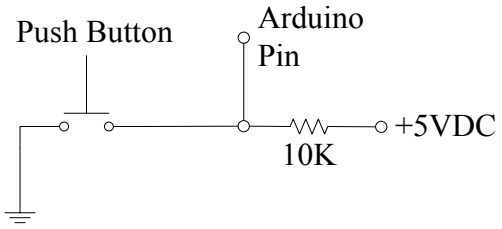

Figure 61: Arduino Digital Input Circuit.

System Health Monitoring

Since the sensor systems are not located in the lab, the system needs to be able to relay information about its functional status to a remote user as well as be able to locally handle some critical error conditions. One aspect of system health is to ensure that all systems are functioning properly. Another aspect of system health is to prevent saturating the data stream caused by critical system errors. A brief explanation of these two health features will be presented.

The periodic health check messages serve as a mechanism to ensure functionality of the system as a whole. First and foremost, these checks provide notification that the microcontroller is running normally. Second, during a health check, the vision system is triggered by the microcontroller's LED. Verifying that there is indeed an event which corresponds to the SMS messaging time stamp allows for full verification of system operation. In essence, this check verifies the complete functionality of the data communication and collecting streams. The time interval between checks is set to 7 hours. Since 7 is not a factor of 24, the daily reporting times revolve throughout the day. This allows the system be checked during all ambient lighting conditions throughout the year.

Another feature of the health check system has to do with detecting a critical error point. The ultimate goal is to prevent saturating the data stream with incorrect data. When a critical error is detected, the system will programmatically disable the accelerometer triggering routine. The flag setting reported by the period health check will indicate that something is wrong. Additionally, both the flag pin and the flag LED will flash. This means that if it is believed the system is in an unhealthy condition, the Raven modem can be accessed through the web for verification. System administrators can log into the modem and look at the states on the digital IO pins on the

modem. If the value associated with the flag digital IO changes over a short period of time, the sensor is locked in the error mode. Another indicator of this is to look at the messaging stream from the Raven modem to see if the messaging pattern seems grossly atypical. In any case, when this occurs, a physical visit to the site is required for inspection and repair.

Ultimately the goal of the health system is to develop a robust system. The two main aspects of the health system discussed above are presented to give the reader a sense of how some of the top level concerns were address by the design. The goal was to provide the user a means by which to remotely verify proper system operation. Additional mechanisms were put in place to help diagnose the system without any traffic exposure associated with a physical visit to the site. By having an understanding of the health aspects of the system presented above as well as a good overall sense of the system as a whole, many potential issues can be diagnosed and repairs can be made which only require a single visit.

Final Circuit Design Schematic

After discussing the functional characteristics of the microcontroller, efforts can shift towards the physical implementation of the hardware. Figure 62 below presents the circuit design. The only feature in this circuit which has not been discussed is a switch which turns "on" or "off" the power bus to both the "Flag" and "Send" LEDS. The reasoning behind this is to help conserve power. Although the LED's use a minimal amount, this will become more critical as the system is further developed in an attempt to make a solar unit as discussed in Appendix C.

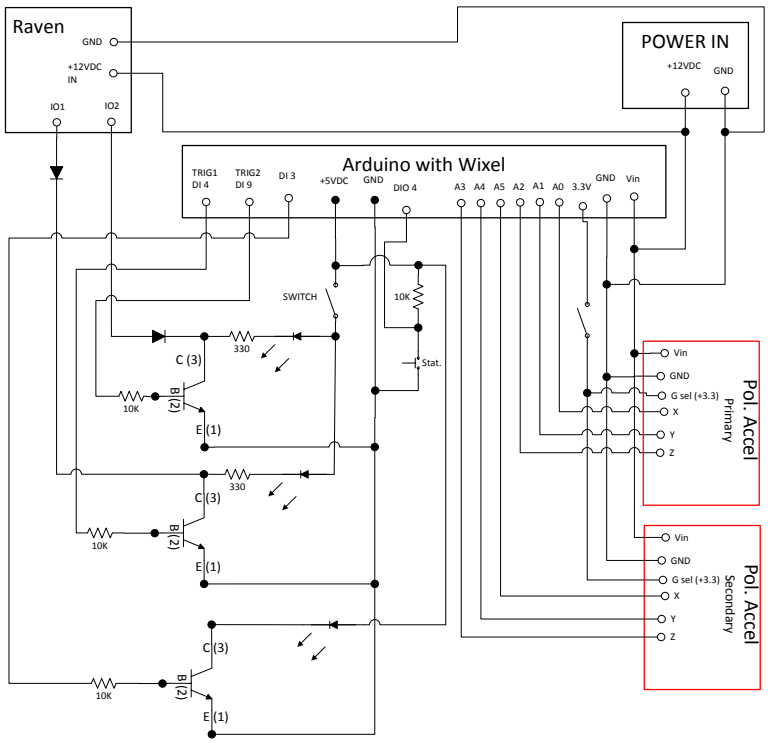

Figure 62: Arduino Box Schematic.

Now that the schematic is outlined, details regarding the fabrication on the Wixel shield will be outlined in a step by step process. The shield is designed to allow for integration with a Wixel

wireless board as well as throughput access to the Arduino's various IO pins. The details regarding the Wixel board's connections to the Arduino will be left to the manufacturer documentation. However, the additional electronics that are added to the Wixel shield that are specific to this application will be outlined below.

The first step in the fabrication is the basic power distribution as shown in Figure 63. The 12VDC flying leads are used to create a +12VDC bus on the board. A two pin header is attached here which is used to supply power to the Raven modem. This was done in the microcontroller box in order to allow for a common cable to be used to interface with the Raven. A continuous 5VDC bus was created to allow for simple component placement. A switch is connected to make a switchable +5VDC bus for supplying power to the "Flag" and "Send" LEDS. The switch was included in order to conserve power when the system is deployed.

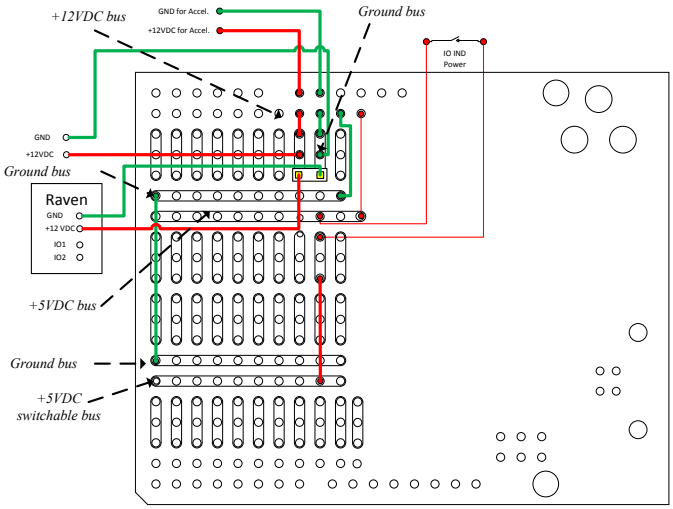

Figure 63: Power distribution.

The next layer of the board is the digital input circuit shown in Figure 64. This is connected to digital IO 2 on the Arduino. When the button is pressed, the digital input is pulled to 0. The physical implementation of this circuit is shown Figure 65 below.

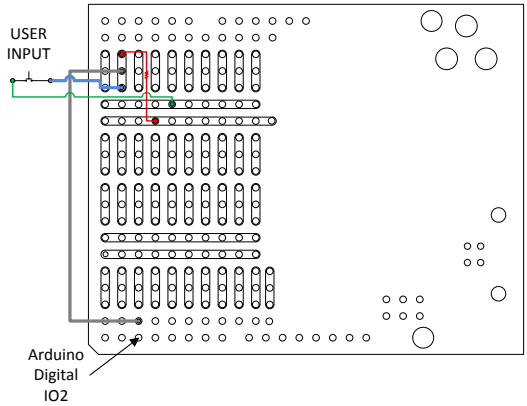

Figure 64: Digital input physical layout.

The camera LED circuit is much simpler than the Raven triggering system. Flying leads come out of the Arduino box that connects to the main terminal strip. Internal to the box, these leads connect to a port on the board. This is connected to Digital IO 3 of the Arduino as shown in Figure 65 below.

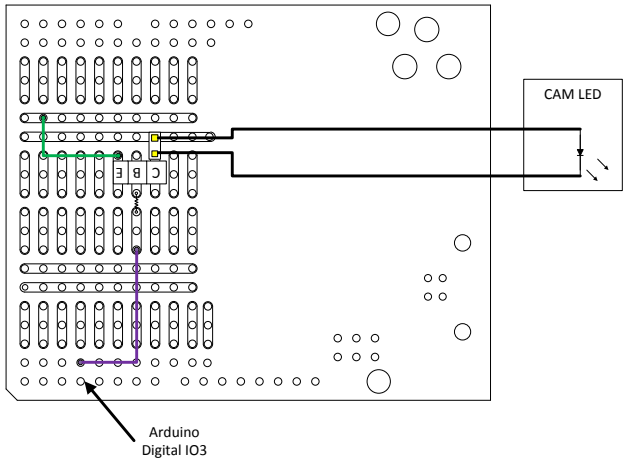

Figure 65: Camera LED layout.

The most complex portion of the system is the Raven's digital IO circuit. This is mainly because of the small amount of room available on the shield. There are three ports on the shield associated with the system. Two of these ports are the two indicator lights for the "flag" and "send" LEDs. They are mounted on the Arduino box. The third port is used to connect to the Raven. The picture below shows the detailed schematic of this system. Digital IO pins 4 and 9 were used for this circuit. The functional schematic of this is shown in Figure 58.

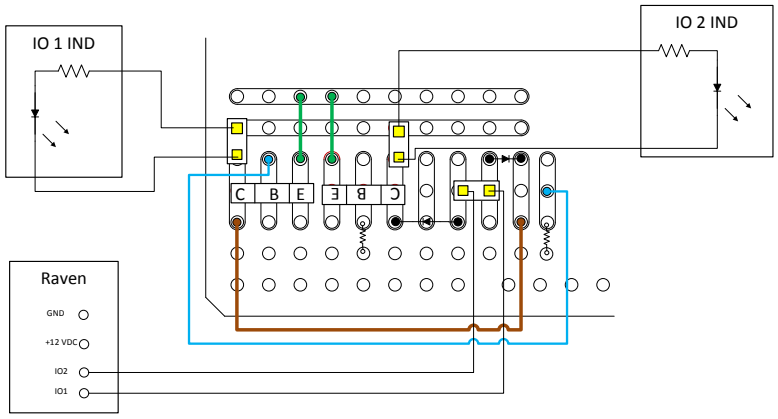

Figure 66: Raven - Arduino interface circuit.

For completeness, the accelerometer connections are shown below in Figure 67. A switch was added to allow the range of the accelerometer to be quickly changed during deployment. These connections go to the accelerometers from the shield's header through a 9 pin CONXALL

bulkhead connector. The G-sel switch is mounted in Arduino box the box and is soldered inline to the wire that connects to the header on the shield.

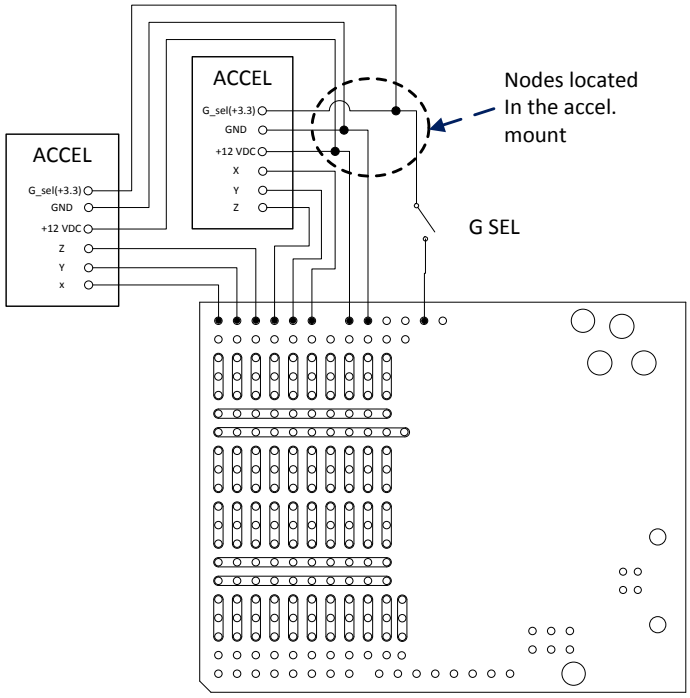

Figure 67: Accelerometer circuit.

Figure 68 below labels the key interface ports that are internal to the Arduino box. These interfaces are used in order to simplify replacing the shield in the event of failure. The Raven Power and IO ports are routed to the 4 pin connecter at the back of the Raven Modem. The Digital IO indicators are connected internally to LEDS that are mounted on the box. The Camera LED pins are attached to flying leads which are used to control the auxiliary LED trigger.

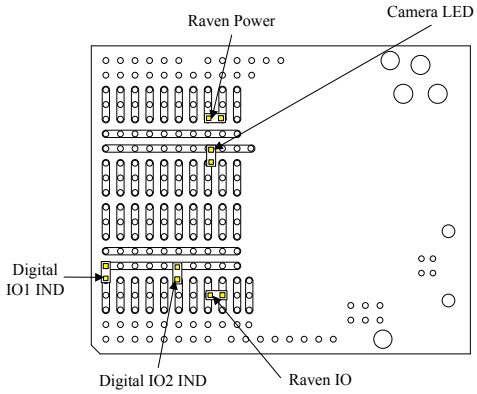

Figure 68: Port layout.

Firmware Development

The Arduino programming environment is fairly easy to understand. The difficulty is in developing the software in a concise manner. The first part of the code consists of variable declaration. These variables are the ones which are used globally. The next section is the "setup" section which is executed once when the microcontroller initially starts. The last key section is the loop section which runs continuously while the system is on. The looping section is portion of the micro-code that is responsible for the bulk of the data processing. The most important part of the code to understand is the looping portion of the system.

A brief explanation of the loop monitoring section of the code will be given without going into the specific details. The first part of the code looks at the 3 axes continuously to detect an event. Once an event is detected, the system continues to monitor the sensor for a short time in order to see if any of the other axes are tripped. This is done to allow for the triggering condition to be met by more than one of the analog inputs. This information could be used to define a direction to the event which may prove useful in the future. Currently, this information is irrelevant as the sensor's directionality was not considered vital at this time. After the loop monitoring section is complete, the controller flashes the camera LED in order to trigger the camera's motion detection software. Once this is done, the microcontroller triggers the Raven modem to send a message. The three message pattern that is generated by an event trigger aides in quickly discerning impacts from the periodic health checks which consist of a single message. These event triggered messages provide a time stamp to the event which can then be used to correlate the exact video along the cameras timeline. Triggering the camera first helps to ensure that a video with the associated event is saved with the maximum amount of pre-event video possible. This would be less of an issue if the user had control over the duration of the captured video, but this is a limitation of the camera's user interface. In this configuration, the time stamp of the message system will be slightly after the associated clip, but close enough to quickly identify the associated clip. A timeline showing this sequence is shown in Figure 69 below. After the messaging sequence is completed, the system waits for a few seconds and then redefines the baseline values for each accelerometer axis. Once these values are redefined, some additional checks are put in place to process information about the specific event internally. These checks are responsible for tracking the overall health of the microcontroller system.

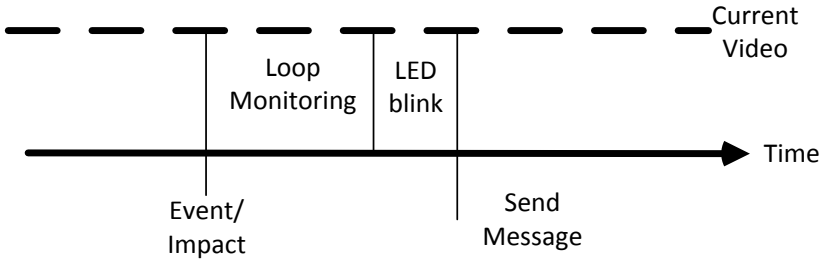

Figure 69: Notification timing.

Electrical System

In summary, the sensor system consists of a camera, modem, IR Illuminator, microcontroller, and accelerometer all of which have been previously discussed. This section will outline the

electrical interface. The physical hardware will follow. Figure 70 gives a schematic of the electrical system. The dotted boxes outline all the external interfaces to the main sensor enclosure. The outline labeled as "Mast Mounted" contains the functional end of the sensor located behind the attenuator. This consists of the camera, IR illuminator, external camera motion trigger, and the antennae for the modem. Another dotted box is intended to outline input power to the enclosure. A 12 volt system was developed which allows for running low voltage power on the ground without conduit according to state regulations. This greatly simplifies system deployment as there is no trenching required. The last dotted box refers to the small accelerometer package. This attached to the attenuator support structure which allows for a physical input to the system as previously mentioned.

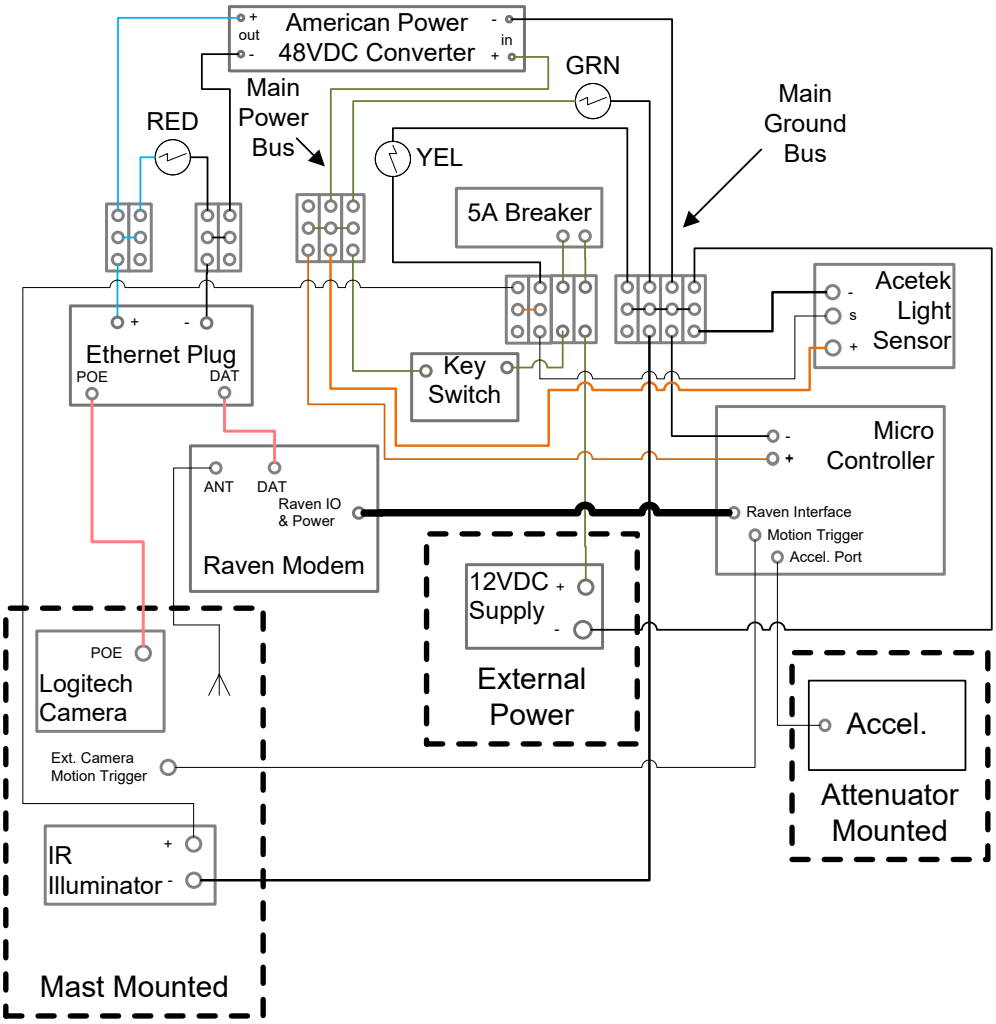

Figure 70: Basic wiring layout.

In order to understand the electrical aspects, a brief functional outline of the wiring schematic will be made. The negative side is connected in the box to set up a ground bus. The positive side of the power source is connected to a 5 amp breaker. The breaker connects to a key switch which functions as an on/off switch for the system. Once the key switch is activated, a 12VDC bus is

turned on (indicated by a green LED located inside the main enclosure). The 12VDC bus provides power to the light sensor, modem, microcontroller and the 48VDC converter. The 48VDC converter is used to provide the proper voltage for the camera's POE and will be indicated by the red LED. The last bus that is created is the 12VDC bus associated with the light sensor. At night, this sensor will switch on power to the illuminator and will turn on the yellow LED. The green, yellow and red LED's are internal to the case and can only be seen when the NEMA box is open. The LEDs are intended to help facilitate troubleshooting the system. In addition to the internal LEDs, the camera system, modem, and microcontroller all have LEDs that convey information regarding overall sensor's operation. Table 8 summarizes most of the LEDs in the system. The LEDs on the modem were not included in the table and the reader can refer to the Raven's documentation in order to understand their meaning.

Table 8: LEDs and their meaning.

Location	Color	Meaning
General Enclosure	Green	Key switched on, and power available
General Enclosure	Yellow	Night time sensor on
General Enclosure	Red	48VDC bus has power
Camera Bottom	Green	Connected to Internet
Camera Bottom	Purple	Connected to Alert Commander
Camera Bottom	Blue	Connected to Alert Commander and internet
Camera Bottom	White	Upgrading
Camera Bottom	Yellow	Camera in standalone mode
Camera Bottom	Light Blue	Not connected to Alert Commander or internet
Camera Bottom	Blinking Red	Camera boot up failure
Camera Bottom	Blinking Red/Yellow	Memory read/write failure
Camera Face	Red	Ready to record
Camera Face	Blinking Red	Recording an event
Camera Housing	Blinking White	Trigger camera motion
Microcontroller housing	Red Pulse	Send message trigger
Microcontroller housing	Orange Pulse	Message flag on microcontroller (turns on when specific accelerometer axis was triggered)
Microcontroller housing	Orange Blinking	Error detected by microcontroller

Additional Electrical Design Considerations

Field deployment yielded some additional challenges to the system which had to be addressed. The fundamental issue is that the closest available power only comes on at night. However, since the power is off during the day, a battery system that is capable of powering the electronics

during the day was developed. Part of this battery system included the integration of a low voltage disconnect. This is due to the fact that if charge on the battery drops too low, the charger is unable to properly recharge the battery and the system will require field service. It should be noted that the highest power demand is at night when the illuminator is on which coincides with the operation of the only available external power source. A simplified electrical layout that shows the power system is shown below in Figure 71. Some extra components were left in the diagram in order to help correlate Figure 71 with Figure 70 above.

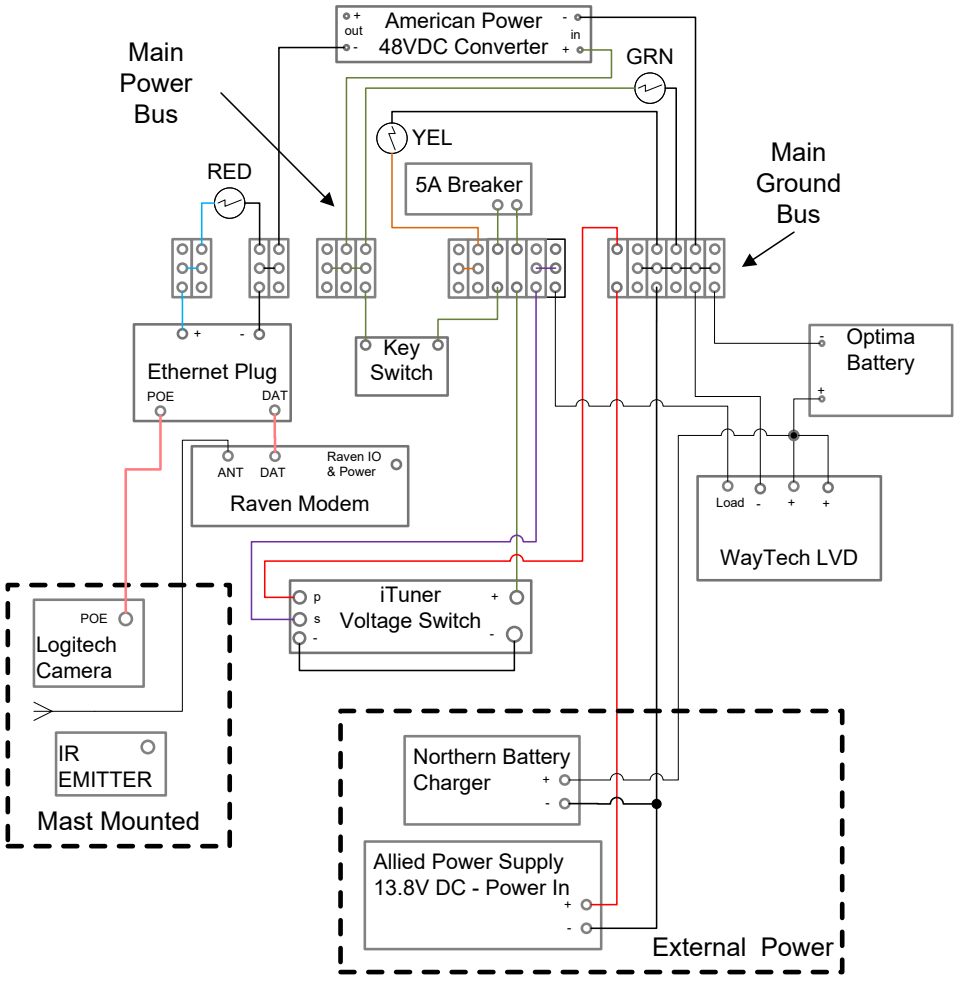

Figure 71: Sensor electrical system.

The key component to understand for the power switching aspect of the system is the iTuner voltage switch (iTuner network PWR-Y-PWR). Fundamentally, this switch compares the input voltages on the primary and secondary pins and pulls power off of the pin with the higher voltage. During the day, the voltage from the Allied power supply (Allied Stk # 70101745) and the battery charger is 0VDC. This sets the iTuner's primary voltage to 0 and the hence the system runs off the iTuner's secondary power that is connected to the battery (through the low voltage disconnect). At night, both the charger and the Allied power supply turn on. Since the Allied supply voltage is greater than the battery charger voltage, the system runs off the Allied

power supply. However, since the positive charger lead is directly connected to the battery, the battery is charged during this time as well.

A mechanism to prevent complete battery discharge was integrated into the power system. If the battery voltage drops too low, the charger in the power box cannot charge the battery due to internal protection in the charger. If this happens, a field visit is required in order to install a fully charged battery. In order to avoid this issue, a low voltage disconnect was added to the system which shuts off the system if the nominal voltage drops below the minimum voltage (which is adjustable on the unit). The modem is the portion of the system which is the least tolerant of low voltage conditions. Therefore the, dropout voltage was set to just above the minimum voltage for the modem. An additional connection was made to the system than what is shown in the manufacturer schematic (shown in Figure 72). The battery charger is connected to terminal strip on Waytech unit where the battery ties into the system. This allows for the battery to charge the system in the low voltage situation. Utilizing this hardware allowed the system to recover from some of the electrical issues that were experienced and caused by blown fuse upstream of the sensors main power box. However, it should be noted that the low voltage disconnect does consume some small amount of power even in the disconnected state. This means that if the power is off for great lengths of time, the battery voltage will still drop below the minimal value for the charger to function properly. This unit was mounted on the door of the enclosure as shown in Figure 73.

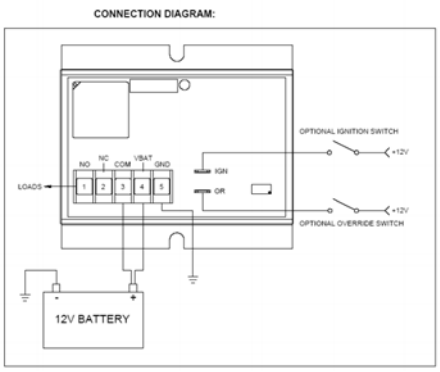

Figure 72: Low voltage disconnect diagram.

Figure 73: Physical Mounting for Low Voltage Disconnect.

The electrical components above were installed in a NEMA enclosure as can be seen below in Figure 74. This gives a general sense of the physical components and their placement within the sensor box enclosure. The only item not included in this image is the microcontroller which is shown in Figure 80.

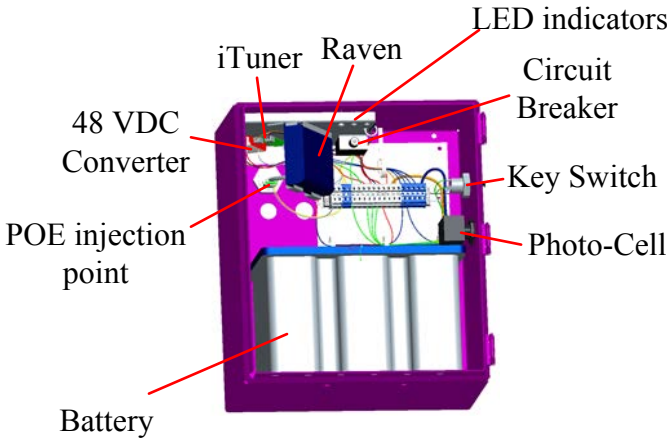

Figure 74: Component layout of NEMA enclosure which is also referred to as the pole box.

Both the Northern Charger and the Allied Power Supply are connected within the pole box to a step down transformer which supplies both the charger and power supply with 120 VAC. The ground terminals of both devices are combined to into a single connection that feeds into the 3 conductor outlet on the pole box. This box was designed to allow the systems to be deployed in two steps. Step one was to install the power box and verify the correct power coming out of the box. This required interacting with Caltrans electricians to connect the system to their electrical grid. Once that step was done, connecting the electrical to the sensor box is simply a matter of plugging the NEMA enclosure into the power box during the installation of the physical hardware that is located adjacent to the attenuator.

Physical Hardware Development

Now that an outline of the electrical system has been made as well as an overview of the system components, a brief description of the physical hardware will be presented. In order to facilitate the deployment, efforts were made to keep all the hardware mounting of the various components as flexible as possible. This simplified the development of any site specific mounting hardware as well as allowing for system optimization in the field. Field optimization was centered on being able to quickly adjust component placement in order to optimize the camera's field of view once installed. Figure 75 outlines the key functional components of the system which are the mast, camera mount, illuminator mount, antennae and sensor box.

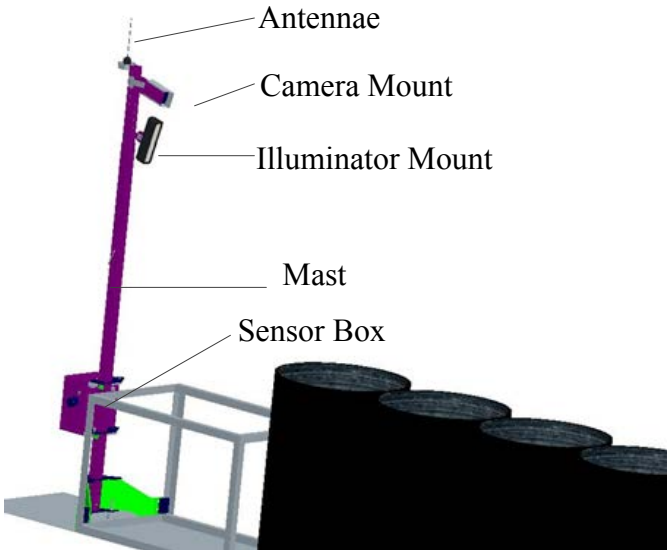

Figure 75: Generalized sensor system.

Mast

One variable that is important to adjust is the sensor height. In order to allow for this flexibility, a telescoping tube system was used. A heavy duty telescoping tube frame from McMaster-Carr formed the system's mast. The perforated series was used in order to give vertical flexibility to the system. The camera and illuminator can be positioned at various locations with a resolution of 1.0 inch. Tubes of different sizes can be nested and locked in place by either using a vendor supplied pin or using a nut and bolt connection.

Camera Mount

The camera mount is adjusted in two ways. First, the camera mount is designed to be able to rotate relative to the mast. This was done by incorporating circular slots in the camera mounting plate. This can be seen in Figure 76 below. The orientation degree of freedom allows the camera to be rotated in such a way to allow for optimal alignment to the attenuator with respect to the direction of traffic flow. In order adjust the camera's field of view so that the attenuator is centered in the field of view, the mast interface bracket was bent (This was later upgraded with a pivoting bracket). Although this method of adjustment is very crude, it only needed to be done once. The drawback to bending the bracket was that the camera mount is now specific to the site and no longer a general sub-assembly. The mobile application facilitated installation by allowing adjustments to be made quickly in the field with immediate feedback.

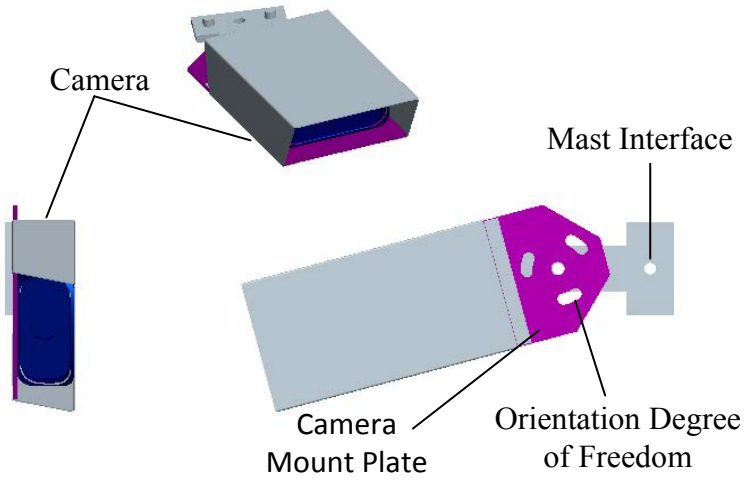

Figure 76: Camera mount.

A small bracket was also created to support the auxiliary LED trigger within the camera's field of view. This bracket was designed to keep the LED as close as possible to the camera in order to prevent headlights from triggering this system. This LED is mounted in an area that is not part of the region of interest in the camera's field of view as shown in Figure 77. The effect of this LED on the user defined motion zones is illustrated in Figure 78 below.

Figure 77: Physical camera LED mount.

Figure 78: Camera FOV with LED.

IR Illuminator Mount

The Illuminator mount was designed for maximum flexibility as well. This is illustrated in Figure 79 below. First, the bracket consists of circular plate which allows for adjustment in the pitch plane of the car (similar to the camera mount). The illuminator is designed to pivot on an axis that is parallel to the lens face. This allows the illuminator to be optimally positioned with respect to the attenuator's centerline and the camera's field of view. Since the system mounts to the perforated tube, the height can also be adjusted. This adjustment is needed in order to orient the illuminator in a way which creates large enough saturation fields in order eliminate some of the shadow triggering. Since there is no control in the camera interface that allows the user to force the camera into night mode, this adjustment was made through a trial and error process which required multiple visits to the various sites.

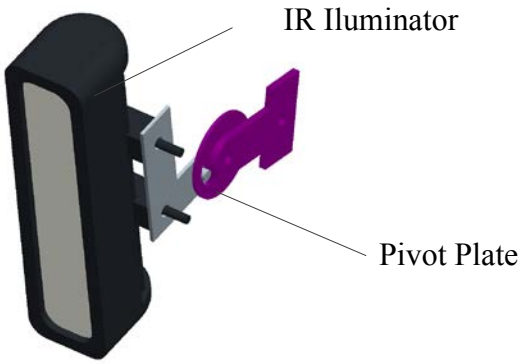

Figure 79: IR illuminator mount.

Antennae

The antennae mount was also a generalized system. This mount placed the antennae at the top of the mast. The key functional aspect of the system was that it provided a more robust strain relief for the antennae cable. This was done to help re-enforce the existing strain relief to reduce any fatigue issues that could arise in the field. The antennae has a magnetic base, therefore the antennae mount was made out of a ferrous metal to utilize this inherent feature of the product.

Sensor Box

The sensor box consisted of a NEMA enclosure (Figure 80). All the electrical connections of the system that are outlined in the schematic presented in Figure 70 are made in this box. The box was sized large enough to hold the battery used to run the system during the day. All enclosure penetrations were made in a manner that prevents moisture intrusion into the system. The sensor box has a total of 7 water proof penetrations. A network port is provided to connect to the camera. A four pin bulkhead connector is used to connect to both the IR illuminate and the auxiliary camera trigger. Separate watertight cord grips are used to feed power into the system, route the antennae cable, and feed the accelerometer cable into the system. A penetration is required for the key switch used to turn the system on and off. The last penetration was for mounting the optical sensor.

The microcontroller portion of the system is packaged in a single box. This was done in order to maintain a level of modularity in the system in the event of system failure. The final design of the sensor box package is shown below.

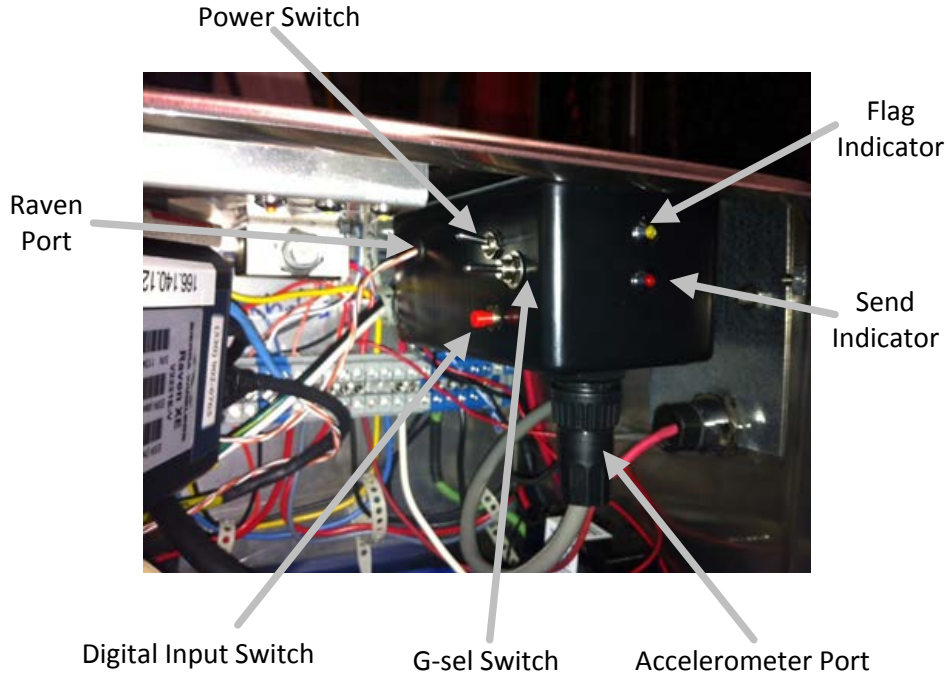

Figure 80: Sensor box with Arduino module installed.

In addition to the interface points labeled above in Figure 80, there are four conductors which could be considered the interface between the Arduino box and the main sensor terminal strip. Two of these conductors are positive and negative leads for the camera LED. The other two are the +12VDC and ground which supplies the power to the Raven (through the Raven port), the Arduino (internally), and the accelerometers (through the accelerometer port).

The LEDs in the front of the box are used to aid in the development of the hardware. The idea was to allow for visually understanding the triggering system as the software was developed. This also allowed field testing the system without the use of any wireless connectivity.

Accelerometer

The accelerometer connects to the Arduino box through a single connector. The idea was to make the accelerometer a modular component to the system. The key interfaces for the accelerometer are the mounting system and the electrical interface.

The mounting system for the accelerometers is fairly simple (Figure 81). The accelerometer units enclose two accelerometer modules. Spacers are used to keep the accelerometer modules away from the metallic housing material in order to prevent any shorting. A single port is also cut into the system that will feed into the sensor box.

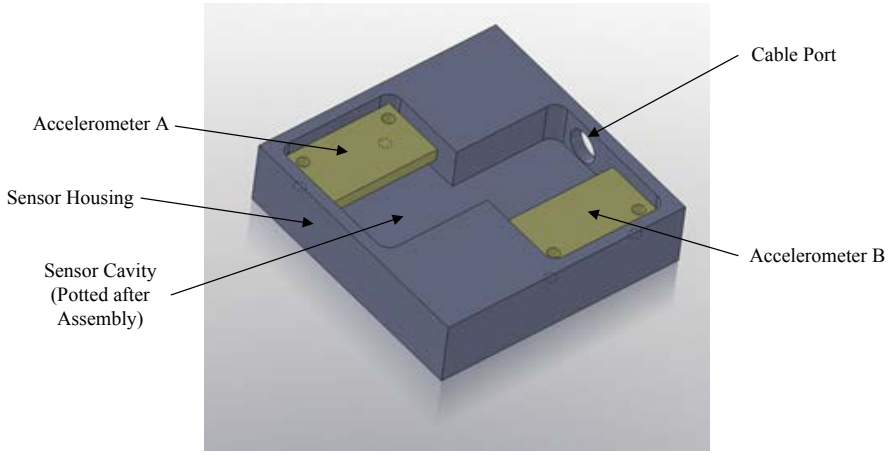

Figure 81: Magnetic accelerometer mount.

In order to keep the electronics protected from moisture, a potting material was used. This helped seal the sensors from water. Additionally, a cover was fashioned for the mounting in order to help further protect the sensors from physical debris.

There are two different versions of the mounting system depending on what product it is going to be installed on. Both the compressor and the Smart Cushion consist primarily of steel components. A case for the accelerometers was made which allowed for the sensors to attach to the steel structure magnetically. This was done by milling out the entire sensor housing out of a solid magnet. Bench top testing showed that the magnetic field did not affect the accelerometer performance. A steel plate was cut to protect the system from any additional damage due to flying debris. This assembly is shown in Figure 82 below.

Figure 82: Magnetic accelerometer mount with cover.

The React 350 required a different design. The idea here is that all impacts will result in a jolt along the cables. Therefore the mounting design for a React has tabs built into the mount which can then be strapped to the guide cable on the system. This will be attached far away from the functional portion of the system in order to prevent any interference. This mount was fabricated out of a solid piece of aluminum and in is shown in Figure 83 below.

Figure 83: React mount.

Another key component was the electrical connections to the Arduino box. In order to make the system module, a single connector was used to connect the accelerometer to the Arduino box. A CONXALL 9 pin connector was used. The Connector pinouts for the cable are shown in Figure 84 and the accelerometer connections are given in Table 9 below. One can appreciate that pins 3, 7, and 8 each have two connections. This connection is made in the body of the accelerometer mount and covered by the potting material.

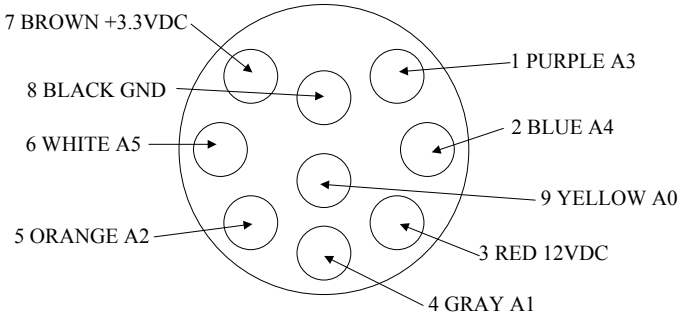

Figure 84: Accelerometer cable pinout.

Table 9: Accelerometer cable pinout description.

Accelerometer 1 Pin	Arduino Box Pin	Conductor Color	Arduino Pin Connection
Vin	Pin 3	Red	12VDC
GND	Pin 8	Black	GND
GSEL	Pin 7	Brown	3.3VDC
X	Pin 9	Yellow	A0
Y	Pin 4	Gray	A1
Z	Pin 5	Orange	A2
Accelerometer 2 Pin	Arduino Box Pin	Conductor Color	Arduino Pin Connection
Vin	Pin 3	Red	12VDC
GND	Pin 8	Black	GN
GSEL	Pin 7	Brown	3.3VDC
X	Pin 1	Purple	A3
Y	Pin 2	Blue	A4
Z	Pin 6	White	A4

Trouble-shooting

Completely understanding the system's functionality is important to troubleshooting the system. The section below documents some of the best ways to diagnose problems with the system which were identified during the deployment support of the systems.

Power issues have been the most common for the system. The system switches between two power systems. During the day the system is battery powered, and at night it is running off of the main power grid. If the system is not working, the power can be checked remotely by trying to communicate with the Raven modem using the fixed IP address. If the Raven cannot be reached, odds are there is a significant power issue in the system. This can be done both during the day and at night in order to get a sense of which power supply system is having the issues. If the modem is unresponsive in both cases, odds are the issue can be attributed to something past the main power box and is caused by an issue with state power grid.

During this project, the Logitech website seemed fairly unreliable. If the camera cannot be reached through the Logitech website, check the modem as described above. If the modem works fine, the odds are that there are issues with the website, and the system should be checked at a later time.

If the messaging system is not working, the user can test the line of communication by sending a test message through the ALEOS software. This allows the user to test the messaging stream for the system. This test may give insight as to the operational status of the microcontroller.

The Arduino box is configured to send periodic health checks. The user should be able to correlate between the health check message and the health check video. It is important to occasionally look at the health check video to ensure that the camera LED is working properly. In the event of a critical error, the user can log into the modem through the ALEOS software and see one of the digital IO pins switching states.

Final Sensor Deployment

The final sensor system describe above was deployed on March 27, 2013. During deployment the attenuator sensitivity was tested. The value of delta was set to 80 before installation. The value for the analog read is 0 to 1023 based on the A/D converter. If the middle point of the accelerometer has a value of approximately 512, a delta threshold of 80 would correspond to about 16% of the accelerometer's range. If the range is selected to be +/- 11g, the threshold will be 1.7g. After attaching the accelerometers, the system was triggered by impacting the attenuator far away from the mounted sensor assembly. This was done by "kicking" the system and not using a vehicle. The sensitivity had to be low enough so that vibrations caused by large trucks passing by did not trigger the system. In all three of the systems which were deployed, the preset value of delta seemed to function very well for the sensor system when tested as discussed above. This test was performed at all three sites in order to ensure the system was functioning properly.

One way to check the system during deployment was to change the modem's messaging protocol prior to installation. The impact notification was changed for installation so that the installers would get immediate notification when the sensor was tripped to a pre-designated cell phone. This was extremely valuable when testing the unit's sensitivity as it could be immediately tested in the field. Additionally, the flashing LED in the camera's FOV produces visible light which could be looked at by one person while another person excited the system.

Final Sensor Performance

Within a short time of the final deployment, four impacts were detected. Three out of the four impact dates would have easily been identified by systematically looking at visible system displacement in the captured videos. The person who is monitoring the data may have been able to home in on the specific video. However, there is no means to determine if an associated video would have been detected or not without the auxiliary triggering LED. The fourth impact created an imperceptible difference in the video images, and hence the impact would be undetectable without the accelerometers. However, in all cases, the Arduino sensor was triggered which in turn triggered the camera's motion detection software. This is obvious as all the recorded impact videos show a flashing LED. Fundamentally, this shows that the system operates very well and provides accurate robust data.

One interesting discovery was that during the development of this system, some of the sensors would miss the health check video. It was ultimately determined that this was caused by the camera's internal software. During some lighting conditions the frame rate of the camera drops to such a low rate, that the blinking LED did not register a motion trigger. In order to mitigate this, the motion zone was extended into the skyline of the camera so that the perimeter of the LED's illumination zone, which has significantly more fluctuation in light intensity, would be included. After doing this, the system health check video capture system was highly reliable. Given the nature of the system, only motion zones around the LED are required; however, users may want to include other motion zones as well if there are ever areas of interest as a specific site.

Additional Considerations

The sensor system seems to be a very robust tool that was developed over the course of this project. However, now that the tool is developed, it is important to recognize that this tool could have many other applications.

The sensor system could be used to monitor other areas which experience a significant number of impacts, such as a guardrail which experiences frequent impacts. This would give designers a better sense of the situation in order to come up with solutions to mitigate the issue.

The sensor's microcontroller is easy to customize. Through the course of this project, a robust system to get video information was developed. Although in this situation, an accelerometer was used, other situations could exist which would simply require the accelerometer to be replaced with a different type of sensor. One version of this could be configuring the system such that monitors areas of high copper theft. Since the current system is battery operated, even if the power to the system was cut, there would still be sufficient power to send notifications for a short period of time.

Summary

This section illustrated the detailed design of a sensor system for the crash attenuator project. This sensor system represents a significant tool for the attenuator project. A quick summary of the functional portion of the final sensor system will be summarized here.

The attenuator is monitored by two systems in parallel. The field of view is monitored directly by the system's camera. The system is also monitored by an accelerometer that is physically mounted to the system. The microcontroller works with both the cellular modem and an auxiliary camera triggering system. The microcontroller triggers the modem to send a text message which assigns a time stamp to the event. The LED serves as a trigger mechanism for the video capture system in order to ensure a video is captured. The text message is routed to an email account to allow for a more robust record keeping. The modem allows for the video on the camera's local memory to be accessed wirelessly. Additionally, each modem was configured with a fixed IP address to facilitate remote access. A generalized flow chart for the core functionality of the sensor is shown in Figure 85 below.

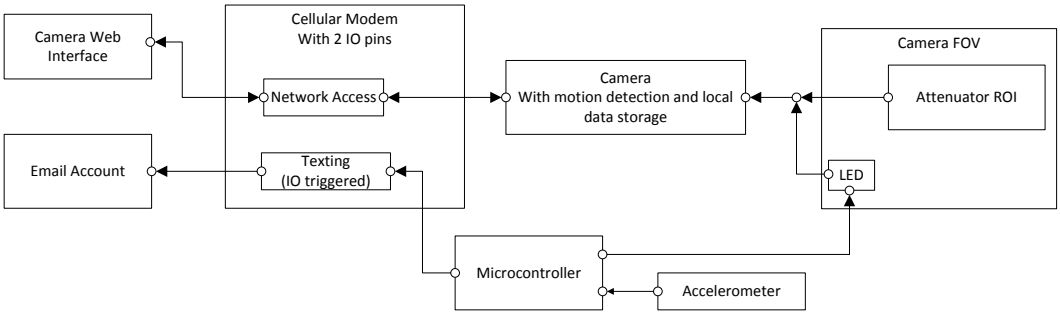

Figure 85: Generalized sensor system flowchart.

The reader should appreciate that many of the features of the sensor system can be manipulated through the web, yielding additional flexibility and facilitating troubleshooting. First, modifying the camera's motion zones can effectively turn on or off the LED triggering systems. Additionally, the messaging stream can be broken by changing the notification settings within the Raven modem. In terms of being able to effectively modify the software, the sole limitation is that the Arduino microcontroller cannot be re-programmed from the office. However, wireless shield that is integrated into the sensor system allows new software to be loaded up without having to directly connect to the system. This communication requires someone to be near the system, but can be done from the safety of the vehicle. This process is extremely fast.

As mentioned above the sensor system was deployed twice. The initial system did not include the microcontroller system. Impacts were detected by sorting through all of the false triggers which was a labor intensive project. During this phase, only 44.4% of known impacts had a corresponding video. Once the new system was deployed, videos were captured for all know impacts. This clearly shows that the addition of the microcontroller system added a tremendous amount of value. Also, this means that the current sensor system provides very accurate and robust data.

Appendix B: CAL-COST USER GUIDE

Starting the Program

There are two ways in which the CAL-COST tool can be executed. The compiled method allows the user to run the code on a windows computer without a MATLAB license. The second method is to directly run the m-file within the MATLAB environment. The steps to start the CAL-COST Tool are as follows:

1) Straight Executable (Compiled Version)
    a. Find the folder which contains the CAL-COST *.exe file and run it.
2) Source Code (Interpreted Version)
    a. Open MATLAB.
    b. Set the Current folder to the folder containing the main CAL-COST file.
    c. Run the main CAL-COST file ("CALCOST_init.m").

In both instances, the folder that contains the working CAL-COST program must also contain the CAL-COST resource file.

Once the program is initialized, the utility will bring up the message box shown in Figure 86.

Figure 86 CAL-COST initialization message box

The "CAL-COST" button will go into the main CAL-COST utility which will aid in life cycle cost analysis for attenuators. The "IMMS" button goes into the IMMS Post-Processor utility which allows the user to interact directly with IMMS database, in order to get a better understanding of existing data on repairs and frequency of hits at a specific location. Both the IMMS Pots-Processor and the CAL-COST utility can be accessed through running specific m-files ("LCAnalysis_reduce_gui_*.m" and "IMMS_DATA_REDUCER_*.m" respectively) from the MATLAB command prompt. This facilitates independent development of the different modules. The name itself is fairly self-evident. In the current form of the program, the flow of information is from the IMMS Post-Processor to the CAL-COST program. It should also be noted that if the user accesses the IMMS Post-Processor from the CAL-COST file menu, all CAL-COST analysis will be lost. There are additional aspects of the program which are included as a demonstration of possibilities to help take the CAL-COST tool beyond just looking at a category based life-cycle cost into the realm of an attenuator deployment and analysis suite.

CAL-COST tool user guide

Then next part of this report will focus on the main CAL-COST utility. This represents the core of the software. Accessing this part of utility is done by simply pressing the "CAL-COST" button shown in Figure 86 above.

Running the tool

Once the CAL-COST tool is started, a window provides the user with some basic background about the tool. This window is shown below as Figure 87. The point of this window is to provide some basic information about how to use the tool. Since this is merely a demonstrational tool, there are some known operational issues which should be avoided. Efforts are being made to eliminate many of these issues, but at this time it is felt best to inform the user of them. Once the user is done with this window, simply press "OK" to continue.

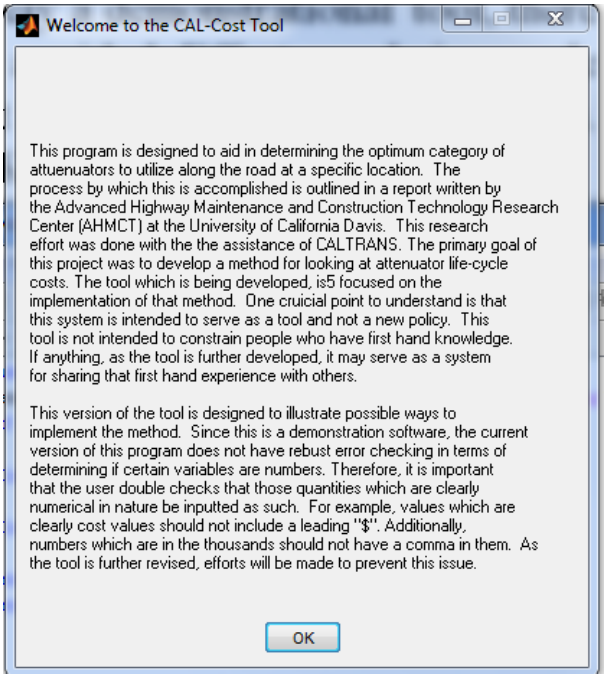

Figure 87 CAL-COST introduction screen

The next window which opens up is the main CAL-COST tool window shown in Figure 88. This panel is the primary user input area of the tool. There are various "panes" associated with this window, serving as a system to logically group various aspects of CAL-COST method.

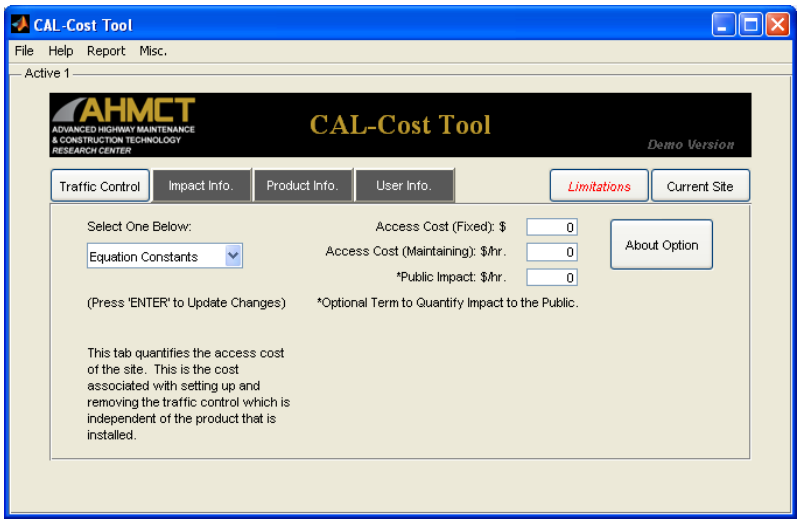

Figure 88 Main CAL-COST window

This window has 3 key areas. The top menu bar area gives the user access to different tool options (which will be discussed later). Below the program title, there is a strip of buttons that constitute different user input sections which are: "Traffic Control", "Impact Info.", "Product Info.", "User Info.", and "Current Site". In addition to these 5 buttons is the "Limitations" button. Although this button is somewhat out of place here, it is critical to make sure that the user is aware of the key limitations to the CAL-COST method. Directly below this strip is the main CAL-COST input window which allows the user to give various pieces of information which aid in estimating the life-cycle cost.

Accessing the life-cycle cost formula

In order to explain the tool use in the most logical fashion, the user menu bar area will be explained in detail later on in this guide. The exception to this is that the menu selection of "***Help > Equation***" will be covered here as it presents the fundamental basis for the methodology behind the tool [19] as shown in Figure 89. This can be opened up at any time while using the tool.

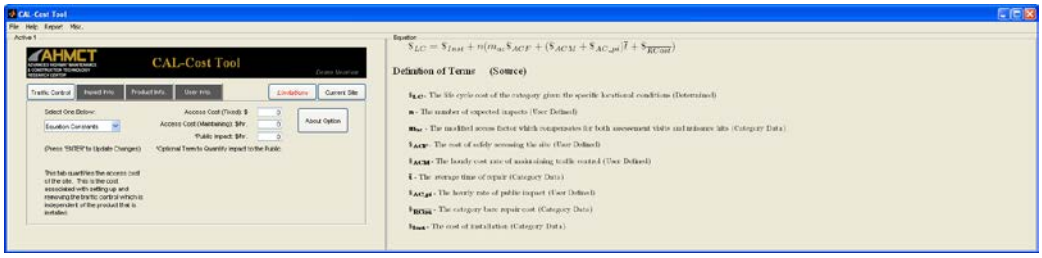

Figure 89 CAL-COST tool with equation

The CAL-COST formula is presented in equation (7) below to give the reader a better sense of the mathematical basis for life-cycle cost analysis.

$$\$_{LC} = \$_{Inst.} + n\left(m_{ac} * \$_{ACF} + \$_{ACM}(\bar{t}) + \$_{AC\_pi}(\bar{t}) + \$_{\overline{RCost}}\right) \quad (7)$$

One of the driving factors behind the formulation of the CAL-COST equation is to decouple the categories' performance from attributes driven by the specific site. Table 10 below gives a brief explanation of the terms in the equation.

Table 10 List of variables used in the life-cycle cost formula

Parameter	Definition	Source
$\$_{Inst.}$	The cost of installation	Category Data
$\bar{t}$	The average time of repair (once the traffic control is in place)	Category Data
$\$_{\overline{RCost}}$	The category bare repair cost (averaged)	Category Data
$\$_{ACF}$	The cost of safely accessing the site (includes setup and removal)	User defined
$\$_{AC\_pi}$	The hourly rate of public impact	User Defined
n	Number of expected impacts (#)	User Defined
$\$_{ACM}$	The hourly cost rate of maintaining traffic control. ($/#)	User Defined
m_{ac}	The modified access factor which compensates for both assessment visits and nuisance hits	Category
$\$_{LC}$	The life cycle cost of the category given the specific conditions of a location	Determined

The reader can refer to the AHMCT report for additional information. This report can be directly accessed by using the "***Help>Tool Basis***" menu selection. The last column in Table 10 gives the reader an indication of the perceived source of the information. The user defined metrics are those which are really location specific and the user must place some value there. The intent is through the collection of robust data, the category costs could be defined as default values. The current tool looks at these values in purely based on IMMS data. By using the tool for a series of case studies, these values will become more robust and realistic.

User input

The CAL-COST tool helps the user compute various aspects of the CAL-COST method. The user input area consists of four main input tabs defined as "Traffic Control" (Shown), "Impact Info.", "Product Info.", and "User Info." The "User Info." tab is only required when the tool is asked to create an output report.

Traffic Control Tab

The traffic control can be characterized by 4 main options which are listed in the traffic control pull down menu: Equation Constants, Basic, Itemized, and Traffic Control Sim. Fundamentally, all of these options are concerned with estimating the fixed access cost ($\$_{ACF}$) and the hourly rate to maintain safe access ($\$_{ACM}$). The four different methods are provided to help the user estimate

the access cost of the site in a way that it is most convenient for them to understand. They were all developed in order to maintain flexibility with how the user must think about access costs

The "Equation Constants" method allows the user to purely assign quantities to both $\$_{ACF}$ and $\$_{ACM}$. This is the simplest method, but may require the user to have the most experience. This is because this method does not provide the user with any guidance to assigning values. The input screen for this method is shown below in Figure 90.

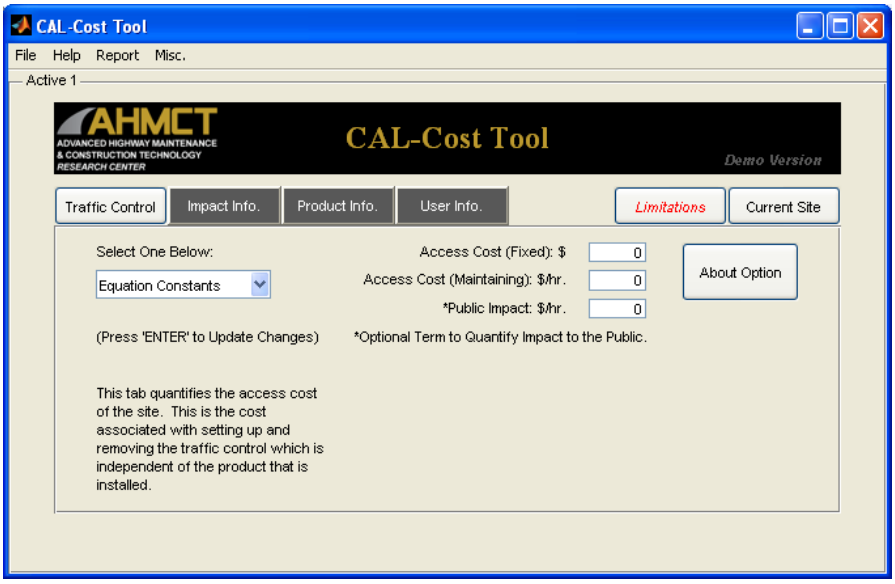

Figure 90 Equation constants - traffic control

The user can get a brief explanation of this method by pressing the "About Option" button. This will bring up the message box shown in Figure 91.

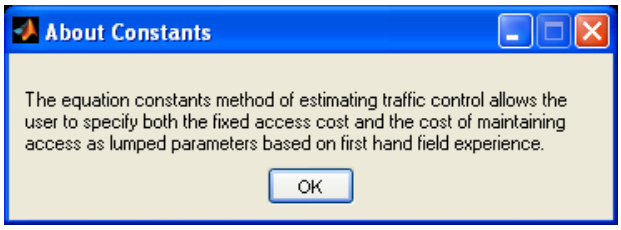

Figure 91 About equation constants

One of the drawbacks to this method is that there is very little justification for the included numbers. Future versions may include a means of explaining the rationale behind the numbers in the form of a user input box.

The "Basic Formula" method allows the user to estimate the traffic control using the relationships defined by equations (8) and (9).

$$\$_{ACM} = \#_{Equip.} * \$/hr_{Equip.} + \#_{Labor} * \$/hr_{Labor} \quad (8)$$

$$\$_{ACF} = \$_{ACM} * t_{Setup} \quad (9)$$

Where

$\#_{Equip}$ = Number of equipment items
$\$/hr_{Equip}$ = Average hourly rate of equipment
$\#_{Labor}$ = Number of Labor Items
$\$/hr_{Labor}$ =Average hourly rate of labor
t_{Setup} = the time to setup/remove traffic control

One issue with this method is the case where a truck that is used for traffic control setup is directly used in the repair. In this case, care must be taken to ensure that the cost associated with the vehicle is properly categorized such that it is properly reflected in the estimation of the various cost metrics. The statement above is manifested by the fact that the user cannot modify the fixed access cost, but can modify the cost of maintaining access within the tool's environment. Figure 92 shows the traffic control input window for the basic formula method.

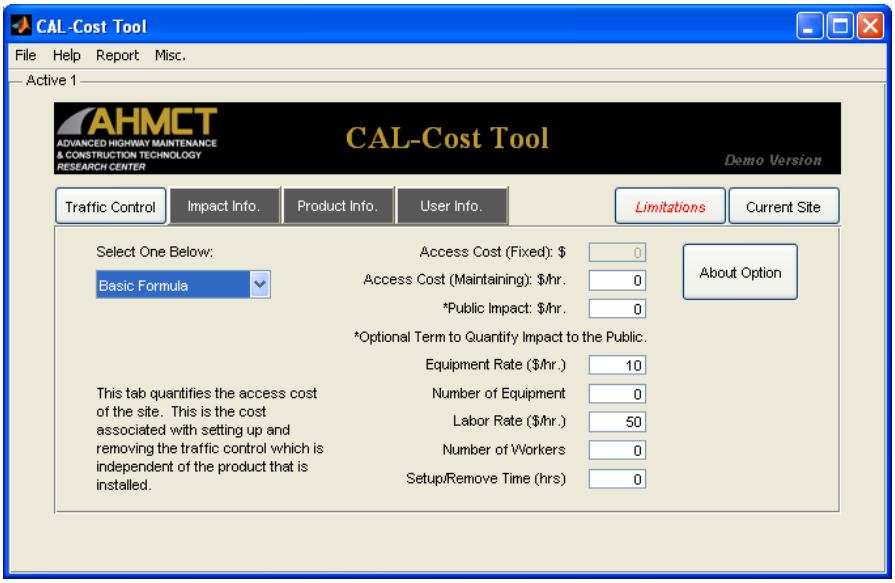

Figure 92 Basic traffic control window

Pressing the "About Option" in this method brings up the window shown below in Figure 93.

Since this method is based on cost rates for both the workers and equipment, the numbers will be easier to follow for anyone looking over the output report in detail when compared to the first method which is a lumped parameter model.

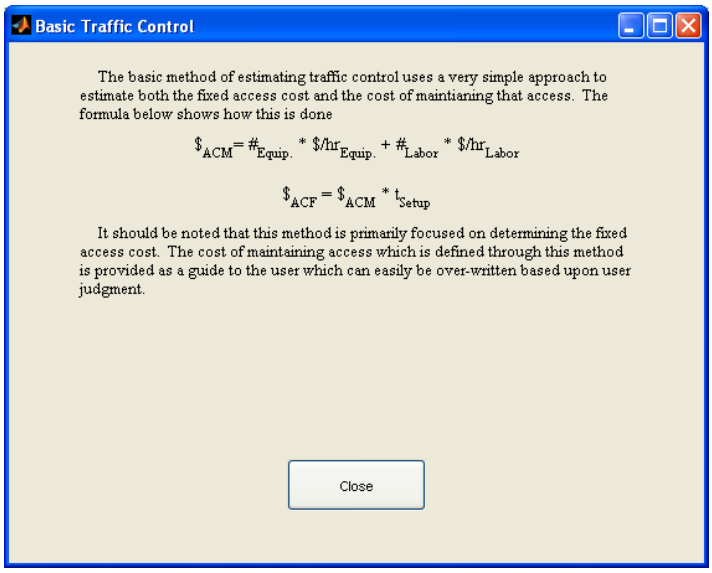

Figure 93 About basic formula

The next method of characterizing traffic control is the "Itemized Costs" method. This method was an attempt to synchronize the traffic control characterization with the Caltrans IMMS database format. Selecting "Itemized Costs" from the traffic control pull down brings up the window shown in Figure 94.

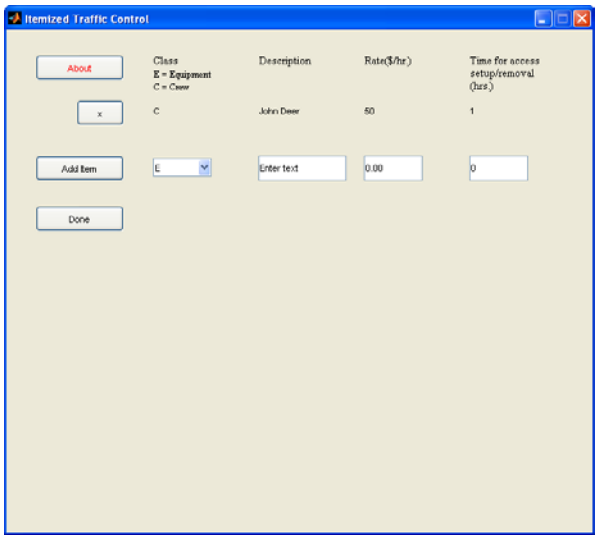

Figure 94 Itemized traffic control window

The "About" button gives the mathematical representation of how the itemized method estimates the traffic control costs, which is shown in Figure 95.

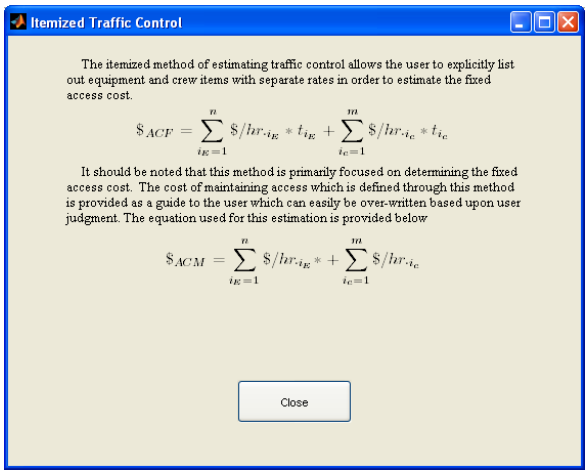

Figure 95 About itemized traffic control

The user must fill out the information for a specific item by assigning a name, cost rate, time required, and a Crew ("C") or Equipment ("E") designation. Once an item is characterized, it can be included in the traffic control estimation by pressing the "Add Item" button. In order to remove an item from the itemized traffic control, simply press the 'X' button. When all items are added, press the "Done" button to return to the main screen. The window will then appear as shown in Figure 96 below.

Figure 96 Itemized costs - main window

This window has a button labeled as "See Items". This button will bring the user back to the window shown in Figure 94 to allow for further editing. Pressing the "About Option" here shows the window shown in Figure 97.

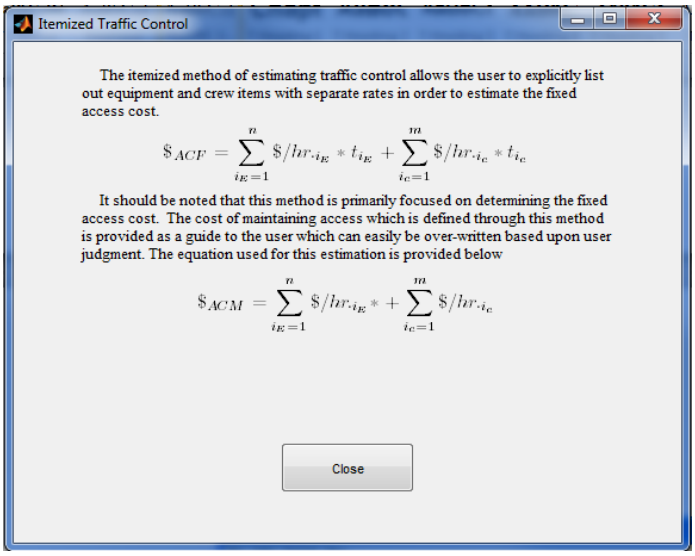

Figure 97 Itemized About Option Button

The last method of estimating traffic control is the traffic control simulator (a.k.a TC-Simulator). This method is a much more elaborate method which utilizes a graphical approach to estimating traffic control. Future versions may allow for use of GPS location of the attenuator to help identify an overview of the site. In its current form, the tool will prompt the user to specify a file which is an aerial view of the site of interest as shown below in Figure 98.

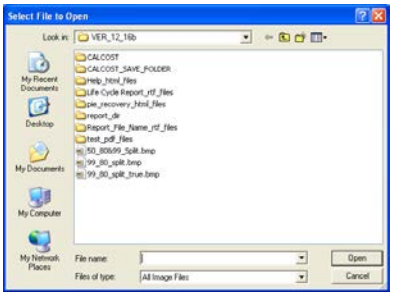

Figure 98 File loading screen

After a file is selected, the CAL-COST tool will bring up the TC-Simulator interface shown in Figure 99.

Figure 99 Traffic control simulator

The traffic control simulator has various interactive areas. The column on the left is used to provide numerical values. These values are the average hourly rates for workers and equipment. Additional key terms are the time of traffic setup. Additionally, this method allows the user to directly state if CHP assistance will be used and asks the user to include the hourly rate for it.

The most interactive part of this method is the ability to make a traffic control plan (TCP) layout on the image above. The current version allows the user to add 5 different types of items. In the future, this could be expanded to develop a site plan to serve as a maintenance guide for performing a repair at the specific location.

One item which can be placed on the image is a marker to identify the target attenuator location. Although this marker is not required in order to estimate traffic control, allowing the user to place a marker for it on the image provides a good reference point for the rest of the traffic control layout. In order to place the attenuator, simply press the “Highlight Attenuator” button. This button will bring up a set of crosshairs which will allow the user to put the attenuator on the image. Once this is done, the tool will prompt the user to identify the item. Typically, this would simply be the name of the current product, but the user is free to label the item as they see fit. Once this done, the “Highlight Attenuator” button changes to the “Replace Attenuator” which will delete the currently placed attenuator marker and allow for placement of a new marker. This item can also be deleted by moving the mouse over the item and pressing the left mouse button and following the prompts.

Four other key elements can be placed on the image. A vehicle or a worker element that is used only for traffic control setup/removal can be added. Additionally, a vehicle or worker element which will be utilized for traffic control for the duration of the repair can also be utilized. Figure 100 below shows an image that has been marked up to represent the traffic control. These items can be removed by either using the “Remove Item” button or by left clicking on the mouse as discussed above. If the user presses the “Remove Item” button, a cross-hair will come up which will then require the user to move the cross-hairs near the target item and then press the left mouse button. The program will then prompt the user for confirmation to remove the item. If the

user does not confirm this selection, the tool will then move to the next closest. This process will be repeated until either the user confirms a deletion or all items have been cycled through.

Figure 100 TC-SIM TCP layout

There are additional features of this layout. The table shown in the lower right corner of the window shows a summary of the items included on the map. Item descriptions within this table can be directly modified by the user. However, in this version of the tool, it is critical that each item is uniquely described in order to prevent operational errors. Future versions will look at eliminating this. The note section in the upper right corner provides the user with a few key pieces of information. The values for the two parameters which represent traffic control are listed here. Additionally, as the mouse moves over the image, a specific item will become larger and the user will be able to see the description of that specific item in the upper right corner. This is designed to aid in modifying/removing items. The calculate button makes sure the values for the traffic costs are up to date. The "Done" button returns the user to the main CAL-COST window.

Once the done button is hit, the user is then returned to the main CAL-COST screen to continue modeling this site. Pressing the "About Option" button here brings up the message box shown in Figure 101 below.

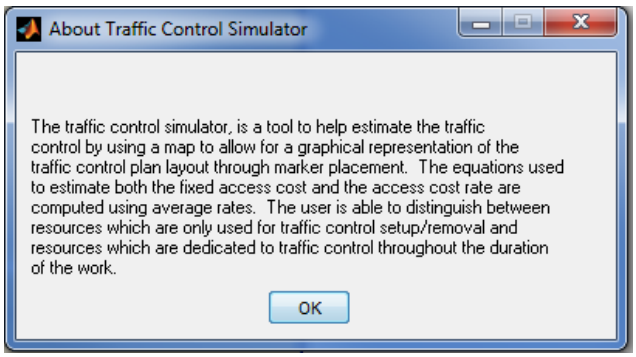

Figure 101 TC-SIM about option button

Public Impact

Another term which is currently defined in each of the traffic control options is the public impact cost ($\$_{AC\_pi}$). The public impact cost is simply a constant value specified by the user which is intended to represent the hourly opportunity cost associated with the loss in the productivity of the general public due to traffic delays during the repair. Not only can this be added in as a lumped rate, there is also a mechanism in the tool to utilize a simple road user cost model.

Another way in which the public impact cost can be characterized is through a simple road user cost model. The premise behind this model is to quantify the cost due to any form of travel delay. This is often referred to as road user cost (RUC). The 2009 travel times cost for delays is given as $12.07/person hour for cars and $29.86/person hour for trucks. In order to come up with the rough estimate for the public impact, equation (10) can be used.

$$\$_{AC\_PI} = \#_{vehicles/hr}\left(\frac{\%_{trucks}}{100}\$_{truck} + \left(1 - \frac{\%_{trucks}}{100}\right)\$_{cars}\right)t_{delay}, \quad \textbf{(10)}$$

Where, $\%_{trucks}$ is the percentage of trucks on the roadway, $\#_{vehicles/hr}$ defines the number of cars, $\$_{truck}$ and $\$_{car}$ are the per person hourly costs associated with trucks and cars respectively, and t_{delay} quantifies the amount of time any individual vehicle is delayed due to the work that is being performed. This simple model relies on the assumption that each car and truck have only a single person.

Impact information

The next primary window is the impact information screen shown below in Figure 102. The range of impacts here determines the x-axis range of the optimal cost plot which will be discussed later. Having the flexibly here allows the user to look at the life-cycle costs on both a micro and macro scale.

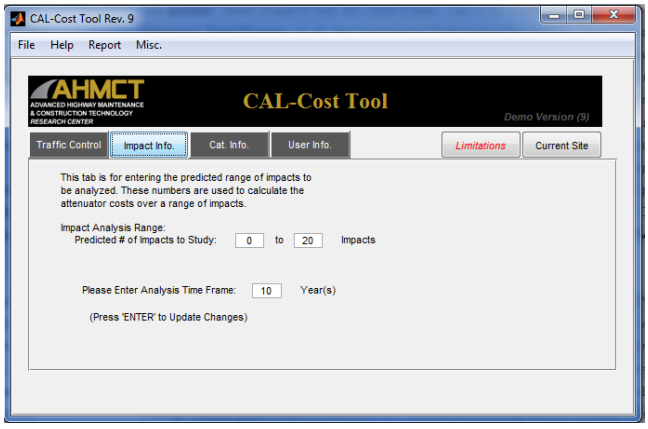

Figure 102 Impact details

The life cycle cost is driven by the number of expected impacts and not a fixed amount of time. However, as the user defines a number of impacts, there is an inherent time frame over which those impacts are expected to occur. During this development of this project, a time frame of 10 years seemed to be the optimal choice. However, like almost any parameter used by the utility, the user has the option of over-riding a value to be more reflective of their experience.

Product information tab

Figure 103 shows the product information tab which is where the user can get additional information about the categories. There are a series of two buttons in this tab. The “List” button allows the user to see key information about specific products within a category. The “Costs” button gives the user access to the average category costs for viewing and editing.

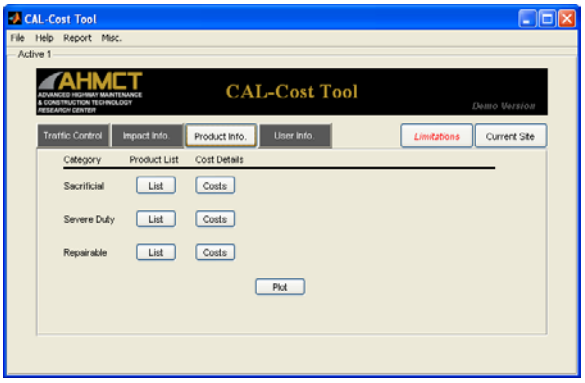

Figure 103 Product information tab

“List” button group

The list buttons give the user access to information about specific products within a category. By pressing one of these buttons, the CAL-COST tool opens the panel shown below. The “Img” button shows an image of the associated product. An additional feature of this panel is a button which links to the manufacturer website for the specific product if the host computer is web-enabled.

Figure 104 CAL-COST example of a product list.

"Costs" button group

The cost button gives the user access to the average category values (Figure 105). These values will be determined by repair cost data that is collected. As more data is collected, this information will be further refined and maintained. This process will help the values to converge on the average value that most accurately reflects the average repair costs. These values are computed from information contained within the CAL-COST resource file.

Figure 105 Category detailed information screen.

The user has the ability to change these values by typing different values into the corresponding boxes (Figure 105). Once the user makes those changes, a window will open up, prompting the user to explain the change to the default values, as shown in Figure 106 below. This reasoning can be edited later and will ultimately serve as justification in the summary report.

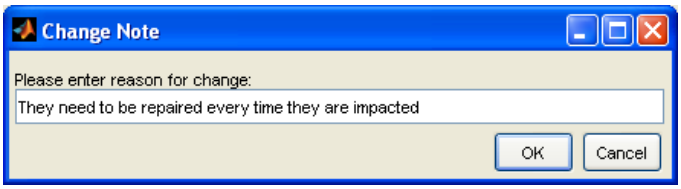

Figure 106 Category data change dialog box.

Once the user changes a default value, that quantity will be displayed in red and a "Default" button will appear as shown in Figure 107. Pressing this button will reset the changed value to the default value..

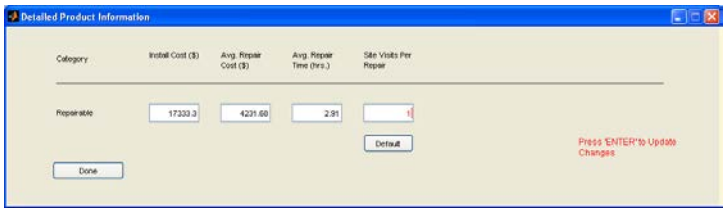

Figure 107 Category details window with change made.

Once the user is done looking at the category values, the "Done Button" can be pressed. This will return the user to the main CAL-COST window. The "*" button shown in Figure 108 below allows the user to view all of the categories changed values as well as the associated justification.

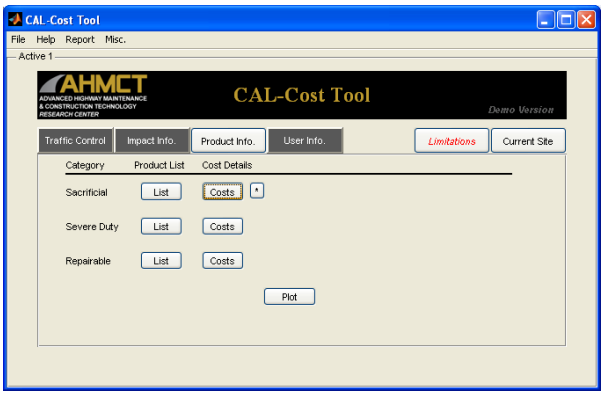

Figure 108 Main screen with "*" button.

By pressing the "*" button, the changes review panel shown in Figure 109 will appear. This widow will allow the user to modify and edit the text that is associated with changes to the default parameters.

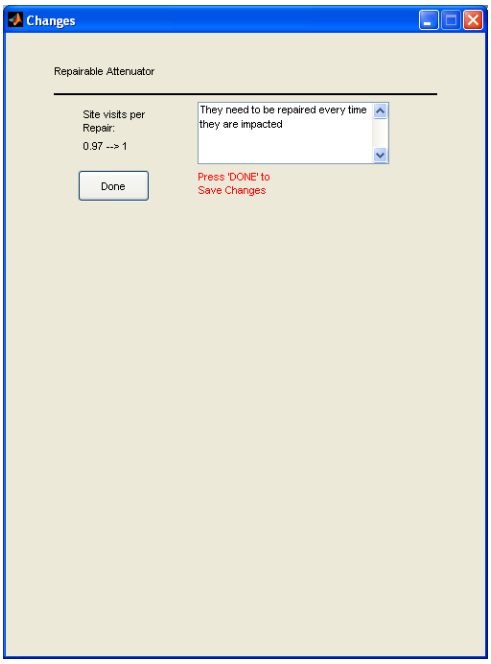

Figure 109 Changes review panel.

When the user is done modifying the text, they can simply press the "Done" button to return to the main CAL-COST screen. Ultimately, any comments here will be integrated into the basic report that is generated.

Plot Button

The plot button listed on the product information tab creates a plot of the "Life Cycle Cost vs. Number of Impacts" as shown in Figure 110 below. This plot graphs the life-cycle cost of all the categories through the range of impacts specified in the impact information tab. Throughout this

report, this will be referred to the "Optimal Plot" as it shows the optimal category to use throughout a range of impacts.

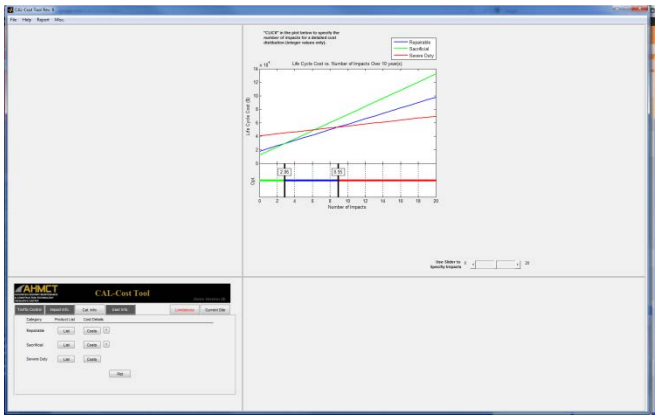

Figure 110 Life cycle cost vs. number of impacts.

Optimal Plot

A brief explanation of the optimal plot will be presented here for completeness. A more detailed explanation is available in [19]. Each category is graphed according to the quantities specified in both the category costs and the traffic control characterization. The program then identifies the crossover points (points where two of the linear plots intersect). Next, the program determines which crossover points are associated with a change in the optimal products and labels those crossover points on the lower graph (indicated by the black vertical lines). The tool then determines what attenuator category is the most cost effective option for a given range. The plot is meant to give the user a category recommendation based on data of the category as a whole and the site characteristics.

Another form of the plot can also be displayed by the utility. An example of this is shown below. The intent is to display the results in a form which is a more familiar way of thinking for many people, but has many limitations in this case. In order to explain this difference, it is helpful to show a more detail image of the optimal category plot form the figure above.

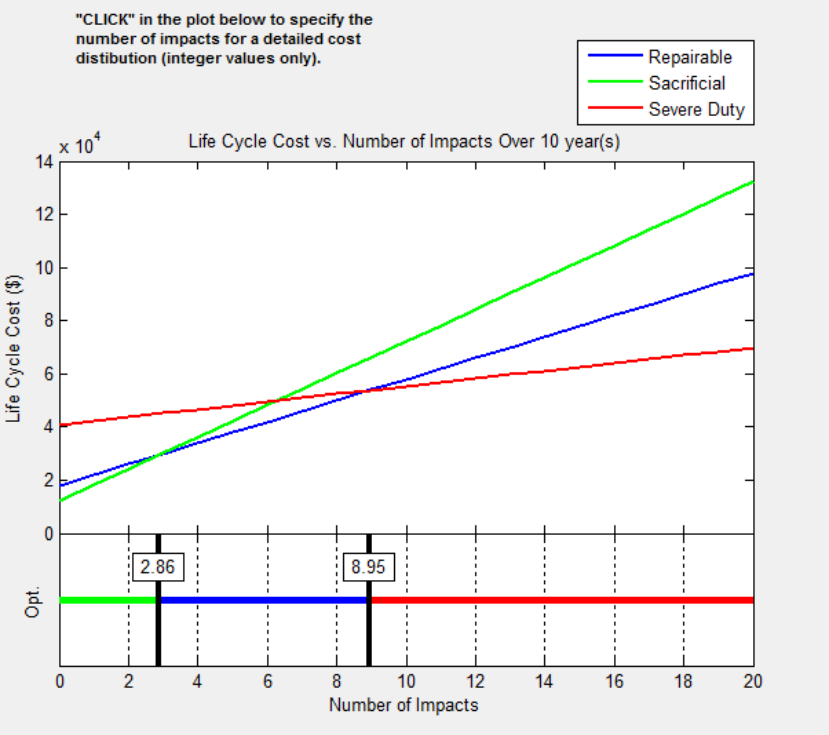

Figure 111 Typical optimal plot.

Figure 111, shows the life cycle cost plotted about the number of impacts. It is important to give the user a sense of time; however, in the plot above, time does not mathematically factor in. The time factor is only meant to establish a basis for estimating the number of impacts. Another way to look at this is to think of the x-axis as the number of impacts per year. This can be done by re-scaling the x-axis in the figure by dividing by the number of years. Figure 112 below shows an optimal plot rescaled to reflect the number of years. The reader can appreciate that the two different plots show identical information, but one plot may have a more intuitive feel than the other.

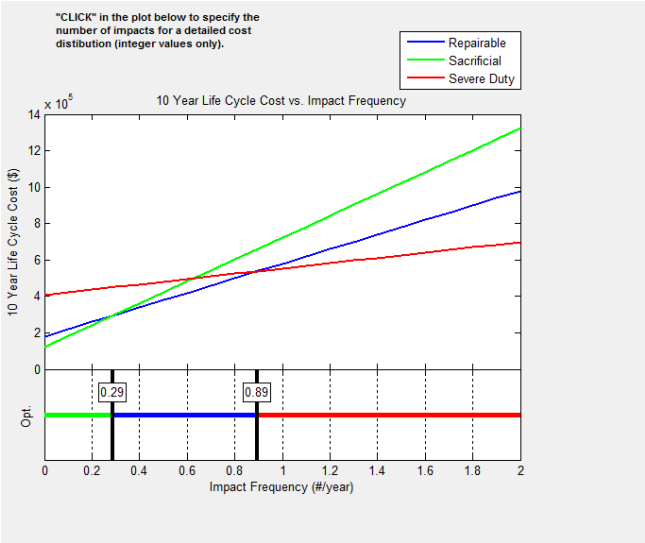

Figure 112 Alternative version of the optimal plot.

The user has the ability to show this plot by selecting the "Change Optimal" option from the "Misc" menu or using the hotkey "Ctlr+D" (D for display). In the current version of the utility, when the plot is shown in this mode, the user cannot specify the number of impacts for showing any pie charts. To go back to the original version of the chart and to re-enable that part of the utility, simply go back to the original plot from by any of the mechanisms mentioned above.

Cost Distribution Plots

Another plot which may be useful for the user is a cost distribution plots. For these plots, the user must specify the number of impacts to look at. This can be specified in one of two ways. The user can "click" in the plot that shows the optimal plot which will pull up a cross-hair to allow the user to specify the number of impacts. The user can also use the slider below the plot to indicate their selection. It should be noted, that the tool analyzes the number of impacts on the basis of an integer number of impacts only since a fractional number of impacts seemed unrealistic. These plots can be seen in Figure 113 below.

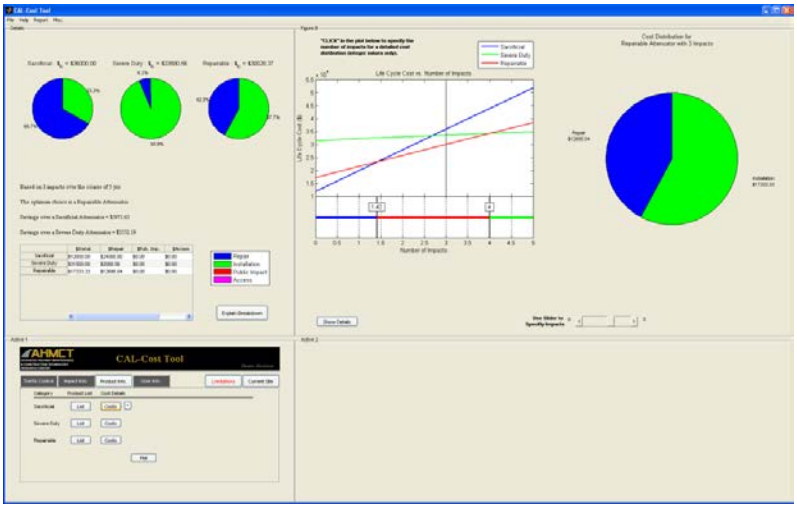

Figure 113 Cost distribution plots.

The CAL-COST tool automatically places the cost distribution of the lowest cost product to the right of the optimal plot. These charts allow the user to see how the cost is distributed. If the tool's recommendation does not make sense to the user, they can investigate the reasoning behind it by studying these charts and reassessing the various terms. The table provided by the tool explicitly lists the numerical basis for the pie charts.

User Information Tab

The user information tab shown in Figure 114, is used for the generation of an output report. This portion of the CAL-COST utility is not required for analysis, but is important to the integrity of the output report. This tab forces the user to provide a direct point of contact for any documentation that is generated by the tool and distributed. This will provide people who are receiving the report to have an easy resource for any additional clarification about the specific situation as well as allowing for clarification of various terms as required.

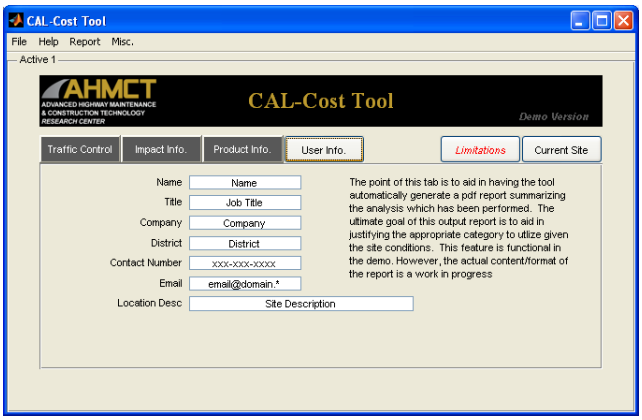

Figure 114 User information tab.

Limitations

The limitations button is intended to highlight many of the other factors that may affect the decision making process, but were difficult to incorporate into the CAL-COST method. Any final output report from the tool will include this list. Since this is a developmental project, users should feel free to contact AHMCT for inquiries to other issues which should be included in this list. The window opens up by the pressing the "Limitations" button shown below in Figure 115. By pressing the "Done" button here, the user is returned to the main CAL-COST screen.

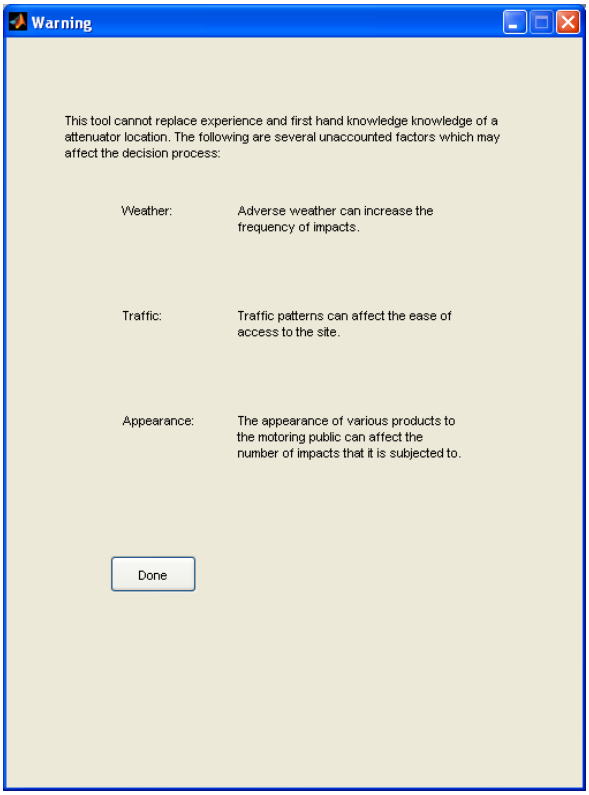

Figure 115 Limitations window.

Current Site Feature

The tool provides the user a means to include the current site information. This is done by pressing the current site button. This will cause the panel shown in Figure 116 to appear. This part of the tool (which is not required) allows the user to catalog previous site information into the tool. The work order number included here is intended to serve as a means for cross referencing any in place record system for the DOT (for example, repairs included in the Caltrans IMMS database have a work order number associated with them). The goal is that information included in this window is not only useful to the specific report which is being generated, but this information can also be distributed to the group which manages the CAL-COST database and used to improve the default values. As data is entered into the green fields, the field changes color to denote that sufficient information is provided. This portion of the code is still being developed. However, some key features will be outlined as they will greatly enhance the tools utility.

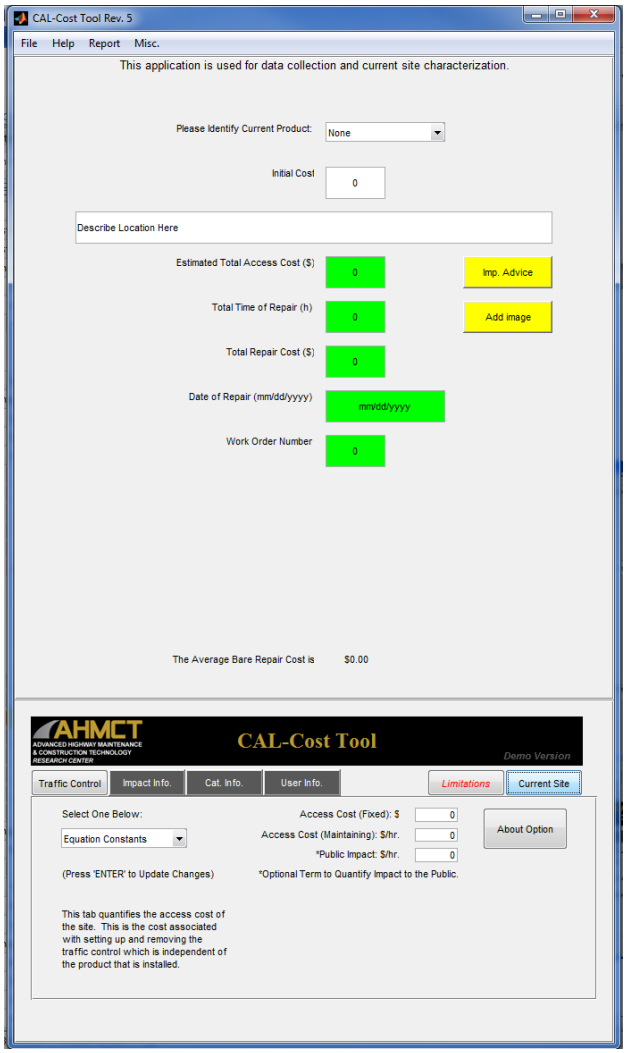

Figure 116 Current site panel.

Once a complete set of data relative to a single repair is entered (i.e. correct data is entered into each required "green" field), the "Current Site" window adds some additional features as shown below in Figure 117.

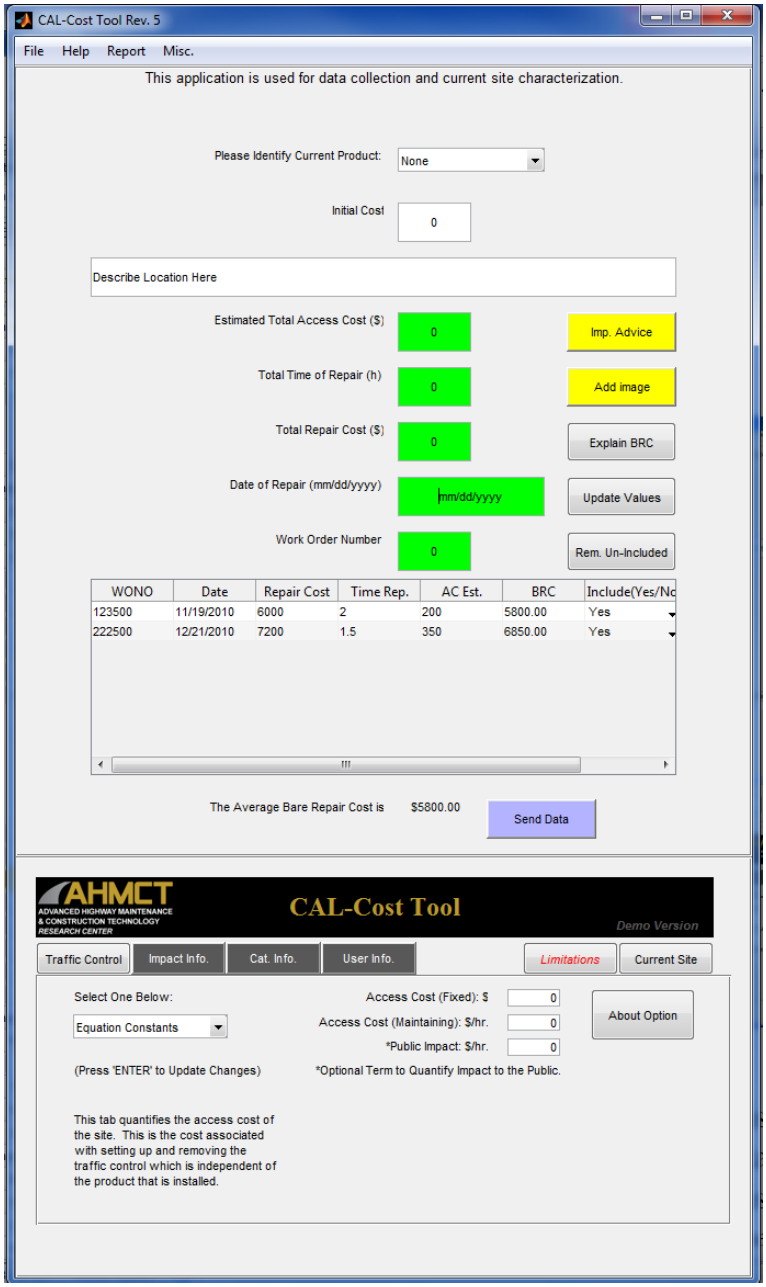

Figure 117 Current site.

There is a table which shows a summary of the information given to utility. The values in this table can be modified in the event that data was incorrectly entered.

There is also a button to include the current site information into the "Life Cycle Cost vs. Number of Impacts" plot. The button will appear after an Optimal Plot is generated. One thing

that is important to understand here is that the tool assumes that the information provided in this window for the total repair cost includes the access cost. The routine estimates the bare repair cost (BRC) based on the access costs that were characterized by the user in the traffic control panel. This process is explained to the user by pressing the "Explain BRC" button. Pressing this button will bring up the window shown in Figure 118 below.

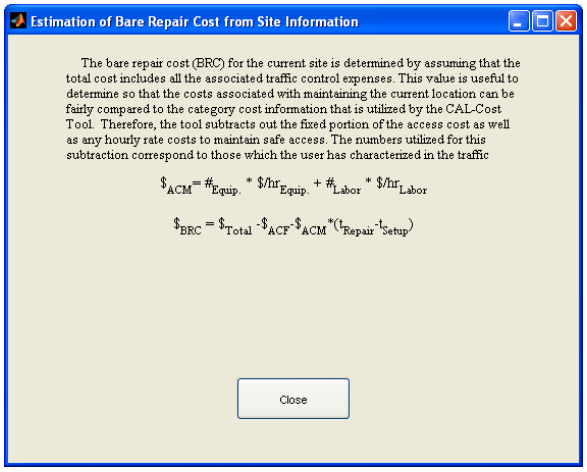

Figure 118 Explain BRC button

The idea behind this process is to allow for an accurate comparison between documented repair costs and the CAL-COST life-cycle cost characterization. This can only be done if the access cost is correctly characterized.

The "Impact Advice" button analyzes the information in the table and provides the user with a recommended range of impacts which the user should consider to implement in the "Impact Information" tab. This button looks at the dates in the table and computes the impact frequency based on the current site information and estimates the number of impacts over the prescribed time frame given in the impact information panel. The reader should recognize that in order to do this process, a minimum of two data points are required.

The last core button here is the "Add Image" button. This image allows the user to select a picture from their hard drive of the local site. This image is intended to give the reader a more direct sense of the sites conditions.

There are a couple of other useful features to this window. There is a horizontal scrollbar at the bottom of the table. The user can move the scroll bar to reveal a column labeled as "Notes". This is an editable field in the table which can be used to provide additional information. The column labeled "Include (Yes/No)" has two different uses. One use is that in some cases the user may wish to completely remove an entry in the table. The user can change the value in this column to "No" and then press the "Rem. Un-Included" button. This will entirely remove a row from the table. The other use of this column is to allow the user to exclude some entries from the computation of the bare repair cost (BRC). The user may want to do that in the case that it is useful to share the data associated with a repair that is out of the ordinary, but does not want that specific repair to be included in the BRC as it is outside what is considered a typical impact.

Once the user makes all the desired changes to this column, they can press the "Update Values" button to update the BRC.

Additional Menu-bar Features

Now that the overall tool features are outlined, it is useful to explain some of the other menu bar features listed in the tool. These items are listed under four key categories which are: File, Help, Report, and Miscellaneous.

The File menu bar section has five options, four of which are fairly standard. The tool has the ability to save and load previous simulations. One current limitation of this is that the tool does not plot the optimal plot/distribution plots that were created in the last simulation. Another option under this menu item is the "Quit" option which is fairly self-explanatory. The forth option here is the "Reset" option. In essence, this option closes and reopens the CAL-COST tool. This is primarily done in the event that there is an error in the tool, and things need to be "rebooted". Using this option deletes any data that has been entered by the user. The last option here is the "GO TO IMMS" option. This allows the user to search the IMMS database for information. However, it should be noted that this option will erase any current work done in the main CAL-COST environment.

The Help menu bar item is intended to give the user access to additional resources for understanding the tool. Currently there are six options under this menu item. The equation button gives the user access to the basic formulation of the CAL-COST method. This is provided to give the user a better sense of what the tool does. The "Glossary" option gives the user access to many of the key terms used by the tool and brings up the panel shown in Figure 119.

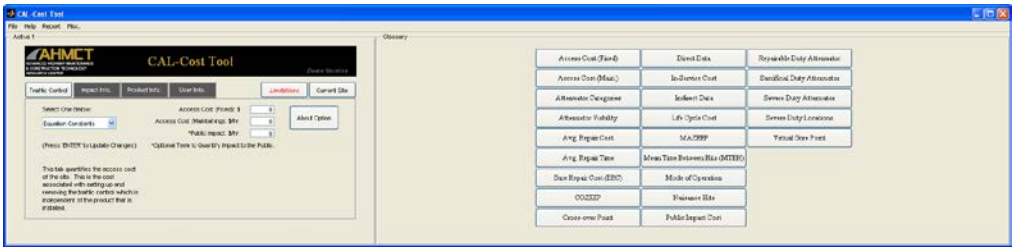

Figure 119 Glossary panel.

Pressing on any button associated with a specific word will open up a window which explains the definition of that term. The Tool Basis selection opens up AHMCT report which presents the logical basis of the CAL-COST tool. This will consist primarily of a compilation of this report and the phase 1 project report. The user guide option will allow the user to open a "*.pdf" file which contains a user guide. Ultimately, this will be an excerpt of this document. This will serve as a means to get access to the information associated with the tools basis. Effort will be made to keep this information synchronized with the latest versions. The "Limitations" option here as an alternative way to access the limitations list (Figure 115). This is intended to help track additional site considerations which the user should be aware of, but are not incorporated into the current life-cycle cost model. A re-plot option is also placed here in order to give the user the ability to update the plots to show the latest values. This is mainly included for debugging the

tool as changing many of the tools quantities automatically results in plot updates. As the tool becomes further developed, it is believed that this menu item will be removed.

The report menu item will allow the user to look at some sample reports that are generated by the tool as well as creating reports from the provided information. The ultimate goal of these reports is to summarize all the information provided by the user and summarize it into a single report. This report can then be submitted as supporting documentation to any Public Interest Finding (PIF) or similar document. Currently, the report is only exported as a word document. Currently, there are some issues with "*.pdf" generation, and future efforts will look at solving those issues. These menu options will not become enabled until the cost distribution plots are made. This is the only requirement to report generation. In other words, the user is not required to fill out the user information tab, include any traffic control values, and include any historical information about the site in order to create a report. With that said, creating a report in this manner really has no meaning.

The last menu bar item is the "Misc." option. Currently, this is not a strongly defined menu group and should be considered a work in progress. One feature of the advanced option is to "enable text move". This option is intended to allow the user to modify the location of text for the various labels for the pie charts. Once enabled, the moveable items will change color and the user can click a specific element and place them at a new location by clicking there. Additionally, it should be noted that when the pie chart annotations are moved, an associated line is created which helps connect the annotation with its associated pie slice. These lines have menu options of their own and will be discussed later. Caution should be made that the positional changes are relatively small in order to keep the right elements next to the correct pie slice. Once the user is done moving the items, the option Advanced>Disable Text Move should be selected to return these annotations to the default "black" color. The user should note that these elements can also be moved directly without using the menu selections. However, it was felt that it would be good to leave this option in the code until the user was familiar with which text elements can be relocated. It should also be noted that any re-plotting will cause the text element to relocate back to the default location and the process described above will need to be repeated. The other option under advanced is the option Advanced> Disable "CLICK". This option disables the user's ability to select the number of impacts to calculate the cost distribution plots by clicking in the graph. The point behind this menu item is to prevent the user from inadvertently changing the number of impacts and hence forcing the utility to update the plots. This also aids modifying some of the graphical plot entities by using an in context menu which is accessible through the right mouse button over the relevant item.

In Context Menu Options

The tool utilizes in context menus to aid in plot formatting. It should be noted that since this is a demonstration tool, some of the formatting changes are lost each time the plots are recreated.

One in-context menu is tied to the slices in the pie charts. The tool allows the user to select the fill color in order to adjust the overall color scheme. This menu option forces all pie items which represent a similar part of the cost in each pie chart to remain completely synchronized. The other option that is included in this method is a way for formatting all key figures in a way that is

generally compatible with black and white printing. This is helpful when creating an output report as any paper copies will be distributed in a black and white format.

Another in-context menu is tied to the linear plots. This menu allows the user to specify line color and line pattern. Additionally, this menu will allow for adjusting line thicknesses in the upper optimal plot. This can be done by selecting the corresponding line in the lower optimal plot. However, this feature does not affect the lower optimal plot. In order to adjust the line thickness for the current site plot, the user must first use the "Disable Click" option from the Misc. menu. As with the in-context menu for the pie slices, there is an option for black and white formatting. When interacting with these elements, disabling the "CLICK" option is highly recommended.

Another in-context menu has to do with the lines that are created when a pie chart annotation is moved. This menu allows the user to move the lines endpoint which is closest to the associated annotation. This menu also gives the user the ability to delete the line completely. The user should be aware that the text movements should be minimal to avoid any system errors. Due to the nature of the code used to create the charts, the line length is limited. This means that even as the user moves text or line endpoints, the final location may not be perfect. Additionally, efforts should be made to make sure that pie chart annotations are relocated to a position that maintains a clear association between the annotation and the associated pie slice.

Tooltip Strings

Many of the tool's user interface objects include what is commonly referred to as "Tooltip Strings". These strings include a brief explanation of the specific interface objects function. This string is accessed by hovering over various tool elements for a small time frame. When the user is done looking at the text, simply move the mouse, and it will disappear.

Hotkeys

Some of the menu items included in the tool have hot keys associated with them which can be access by holding down the "Ctrl" key and pressing the indicated letter. These keys are listed next to the menu option they correspond to. The table below summarizes these buttons.

Table 11 List of available hotkeys in CAL-COST

Ctrl+O	Open File	Ctrl+S	Save File
Ctrl+Q	Quit CAL-COST	Ctrl+E	Equation
Ctrl+R	Reset CAL-COST	Ctrl+L	Limitations
Ctrl+D	Change Optimum Plot	Ctrl+G	Glossary of Terms
Ctrl+P	Generate *.PDF Document	Ctrl+W	Generate Word Document
Ctrl+H	Opens Help File	Ctrol+I	Toggles Impact Selection
Ctrl+M	Toggles text move		

Structure of the CAL-COST Resource File

The CAL-COST utility interacts with an excel file in order to get information regarding the cost averages used by the utility. The excel resource file serves as a means to organize the

information utilized by the utility and the DSS analysis system. Since the data is still in its infancy, direct access to the product specific cost metrics would be inappropriate. This documentation is concerned with the current structure of the file and later versions on the utility may significantly change the overall file structure. However, understanding and documenting the general structure of this file is useful for the incorporation of new data and the addition of new fields. Also, this could aid in giving people involved with any future work a basis for the design of a more complex database.

There are two main types of sheets in the excel resource file. Each product is given its own individual sheet. This sheet contains all the information associated with a specific product. The "Products" sheet contains a summary of all the information that is directly imported into the utility. The over-arching goal of this structure is to have the information organized in such a way that once the data is of sufficient quality, the system could be modified to look at specific products. Currently the data is felt to lack the robustness required to make claims for or against a specific product.

Product Specific Sheets

Each product sheet consists of five main sections. These sections are the Product Information, Data, Matlab Indices, Installation, and Repair sections. The first two are the main part of the sheet used by the tool. The third section contains the indices needed in order to manipulate the data through the "xlsread" and "xlswrite" functions in Matlab. The last two sections are raw information about the specific product repair. In order to best understand the structure of the products sheets, each section will be discussed. However, the MATLAB index section will be discussed out of order as these indices listed here help to define the derived values listed in the "Data" section.

Product Information

The product information section of the excel resource file is intended to contain information that can be directly specified by the product manufacturer. Table 12 below presents a general view of this section of the product sheet.

Table 12 Generalized product information

	A	B	C	D
1	Product Information			
2		Category	Sacrificial	
3		Typical Maintenance Schedule	6	Times/Year
4		Typical Maintenance Cost	500	$/Visit
5		Typical Maintenance Time	0.5	Hours
6		Product Website	http://product_website. com	
7		Product image	Product_image.jpg	

The category designation given in the "C2" field designates the category of the specific product. The utility looks at this flag as a means to organize information by category in order to compute the correct category averages used by the utility.

The fields "C3:C5" are all related to the product maintenance. The current form of the life-cycle cost formula does not currently include maintenance costs. However, fields for this information

were included in order to facilitate the integration of this factor at a later date if needed. The information can be initially defined by the product manufactures which should be part of the general product information provided to end users.

The last two fields in this section are the product website, and a product image. These are used to give the user additional information about the products. This can be accessed through the category list panel accessed through the "Cat. Info." tab (see Figure 104). The image helps provide a general view of the product which will help users be able to visualize the product. The image file must be placed in a folder which coincides with the directory of the executable file. The website, will give the user a means to quickly reference any additional data provided by the specific product manufacturers. This allows the utility to serve not only as an analysis tool, but also as a platform through which to quickly access vendor information for all the products online (if provided).

The Matlab indices section is used to help compute the product averages. This section outlines the rows associated with installation data and repair data. A future module will be implemented into the utility which will allow for assimilation of data into this sheet through the IMMS utility. This is the module which is referenced in Figure 33. Table 13 shows an example of the Matlab indices section. These help to keep track of the amount of data included in the tool.

Table 13 Matlab indicies

	A	B	C
15	Matlab Indices		
16		Installation Start Index	26
17		Installation End Index	26
18		Number of installations	1
19			
20		Repair Start Index	30
21		Repair End Index	71
22		Number of Repairs	42

The installation start index, "C16", identifies the first row that contains information about the installation cost of a specific product. This number will essentially be a constant unless there is a significant change in the general structure of the resource file. The installation end index will help identify the number of installation data points included for a specific product. The number of installations is also computed into the utility. This index is somewhat redundant. The relationship between the three terms given is by equation (11).

$$\text{Number of Installations} = \text{Installation End Index} - \text{Installation Start Index} + 1 \quad (11)$$

The only violation to this is the case where the number of installations is set to zero which would indicate that no installation data is included. In this case, the Number of installations is set to 1 by default to help prevent any internal errors in the code. The repair indices have similar meaning to the installation indices.

In order to better understand how these indices function, a few examples of how they function will be presented. The first example will be to add an installation entry for the specific product

shown in Table 13. The added installation will increase the installation end index by 1 as well as the number of installations. Additionally, the entire repair data section will shift down a row by 1 as well. The updated Matlab indices table would change as shown in Table 14.

Table 14 Updated index table

	A	B	C
15	Matlab Indices		
16		Installation Start Index	26
17		Installation End Index	27
18		Number of installations	2
19			
20		Repair Start Index	31
21		Repair End Index	72
22		Number of Repairs	42

The other way the table can change is by adding a new repair entry. The addition of another repair would change the table by making the repair end index 73 and the number of repairs would now be 43. These indices are used to compute the values presented in the data section of the product page.

The installation section is used to organize information associated with product installation. The row index for this information is defined by the "Matlab Indices" table. The important thing to discuss is the columns that are included. Table 15 below indicates the various installation fields included in the product information sheet.

Table 15 Installation fields

Column	Text	Column	Text
B	Date	C	IMMS
D	Location	E	GPS Lat.
F	GPS Long	G	Supervisor
H	Phone Number	I	Email
J	Data Type	K	Base Product Cost
L	Job Cost Above Base	M	Installation Notes

Many of these fields are currently included for reference only and can only be accessed directly through opening up the excel file. The two most significant fields are the "Base Product Cost" and the "Job Cost Above Base" fields. These fields include information used to compute the average cost of installation for the product. The base product cost is intended to capture the cost of the physical product, and the cost above base is the additional cost incurred through site preparation and product installation. Future efforts will be made to segregate these two aspects of the installation cost in a more rigorous manner.

Table 16 gives a brief description of the fields contained in the repair section of the sheet. The most important fields in Table 16 are the repair cost, number of visits and the repair time. These three fields will be utilized to compute some of the derived quantities that are used by the utility.

Table 16 Repair table columns

Column	Text	Column	Text
B	Date	C	IMMS
D	Location	E	GPS Lat
F	GPS Long	G	Supervisor
H	Phone Number	I	Email
J	Data Type	K	Parts Cost
L	Parts Description	M	Number of Visits
N	Number of People	O	Repair time(hr)
P	Repair Cost (No traffic Control)	Q	Traffic Control Cost
R	Notes about repair	S	Notes about impact

There are some commonalities between the installation and repair data sections of the sheet that should be discussed. First, as technology becomes increasingly available, the integration of GPS information (Columns "E" and "F") into the data seems to become a more feasible option for data organization. This would be a more robust mechanism for associating specific work order numbers to an exact location. In more developed areas, attenuators can be close enough such that the previous system of using post miles to identify a location begins to break down. Column "D" is intended to give a space to include a text description of the specific location. Columns G-I are intended to provide contact information for the person responsible for the repair in order to aid in clarification of any of the information. The contact information could also be used to upgrade the "quality" of a data point as discussed below.

The "Data Type" field is intended to help track the quality of the data. Currently, the data type field is used help build the pie chart shown in Figure 45. Future efforts will be made to help turn the data into a more robust type. This can be facilitated by communicating with the relevant personnel in order to gain a more detailed understanding of the specific repair. This is aided by pieces of information contained in the product sheets, such as supervisor field.

The last section of the product page is the data section. This section contains all the derived quantities used to compute category averages. Table 17 shows an example of this section of the file.

Table 17 Data section example

	A	B	C
8	Data		
9		Average Cost of repair	4300.176679
10		Average Time of repair	4.221904762
11		Access Cost Modifier	1
12		Average Installation Cost Above Base	0
13		Average Installation Cost Base	18062.05

The quantities given in Table 17 are derived values based on the indices listed in the Matlab indices table (Table 13). The average cost of repair is computed by the averaging the repair cost column, "P" from the repair start index to the repair end index. Similarly, the average time of repair, and the access cost modifier can be computed by averaging columns "O" and "M" over the same range. The average installation cost base and average installation cost above base can

be computed by averaging those columns ("K" and "L") over the range defined by the installation start index and the installation end index.

In summary, this section outlined how the specific product data is organized which will allow for utilization in the tool. Efforts were made to explain the relevant excel indices. These indices are critical to the utility. Future efforts will be made in the utility to make the identification of these cell references more robust. However, it was felt that outlining the basic layout of the resource file would facilitate the implementation of any future changes.

"Products" Sheet

After outlining how specific product data is integrated, it is important to outline the "Products" sheet. This sheet is the main resource sheet for the utility. This sheet lists all of information required by the utility. There are three fixed values on this sheet. One is the hourly labor rate that is contained in cell "D6". Another is the hourly equipment rate in cell "D7". The last fixed quantity is the number of included products listed in cell "D8".

The key data about each product is summarized on this sheet. This information begins on row 13 and ends on row 13+"D8"-1. Table 18 gives a brief overview of the information contained here.

Table 18 Product summary information

Column	Text	Column	Text
A	Product	B	Category
C	number of data points Installation	D	Number of data points Repair
E	Maintenance Frequency	F	Maintenance Time
G	$ main.	H	Access Cost Modifier
I	Repair Cost	J	Repair Time
K	Installation Base	L	Installation above Base
M	Website	N	Image
O	Front Width(m)	P	Rear Width(m)
Q	Length(m)	R	Side Gating
S	Side Def.(m)	T	Front Gating
U	Front Def.(m)	V	Number Installed

The CAL-COST utility reads in all of the required information during each execution. The cost metrics used by the life-cycle cost analysis are average values computed from this table by grouping the specific products by category (The "B" column). Columns "O-V" are used for the DSS analysis. Future efforts will be made to shift these pieces of information into the product sheet data section. The decision support system (DSS) utility, which will be briefly discussed later, was built solely as a demonstration tool and data has not been rigorously defined at this time.

Functional Notes about the Excel Resource File

There are a few functional notes about the excel resource file that are important to point out. These notes are important to keep in mind as future changes are made in order to prevent functional errors in the utility as well as incorrect data.

Each product sheet in the excel resource file must be given the same name as the associated product listed in column "A" of the "products" sheet. This will become more and more of a

requirement as the data assimilator module is further developed. This module will have to be able to rely on the product name to identify the corresponding sheet for data manipulation. The embedded excel formulas are updated as new data is integrated.

Currently IMMS data is the only detailed source of information available. In an effort to maintain some form of robustness to the data, efforts were made to prevent duplicate entries of the same work order number. This can be done by keeping track of the IMMS work order number and comparing that with those already entered into the utility. This will prevent the same repair from having multiple entries into the resource file. The current process by which this is done creates some issues. First, the process by which the work order number is checked requires the user to be sure to attribute the repair to the correct product. Also, the comparison process is more of a brute force approach and is very time intensive. It is believed that as the database grows, this process will take even more time until it becomes unreasonable. At that point, efforts will need to be made to refine this process. Also, there is not mechanism for a repair to be integrated into the data sheet for two different products. This could be the case as the IMMS database lacks sufficient detail to consistently identify the product with the repair without any other knowledge about the specific site.

The last real trick to reading and writing data between the excel file and the Matlab environment is to understand the cell correspondence. Each cell in excel is referenced by a column, designated by a letter, and a row, designated by a number. In the Matlab environment, matrix locations are identified by numerical row and column indices. Therefore when addressing a location in the excel resource file, it is important to properly address the column. This is done by converting the column number, i, to its letter equivalent by using equation (12),

$$\text{Column Letter} = \text{char}(64 + i) \quad (12)$$

where “i” is the equivalent column number. For example, if “i” is equal to one, the result of equation (12) would be “A” according to the standard ASCII character map.

Summary

The goal of this section was to outline the basic structure of the excel resource file. This is intended to facilitate integration of data in the future. Robust programming calls for the elimination of what are called magic numbers. Due to the fact that the data is organized in an excel resource file, many of these “magic number” appear within this file. The main point to this section was to shed light to these so called numbers and facilitate any future changes to the utility.

APPENDIX C: Solar Sensor System

Solar Sensor Upgrade

The sensors described in this report relied on the availability of power in order to operate. During the site selection process, this placed a limitation to where the systems could be deployed. In order to remedy this issue, efforts were made to create a solar powered standalone unit. This required changes to both the software and the Arduino control hardware. These hardware changes reduced power consumption during system operation. This system was tested at the ATIRC facility

Solar Power Hardware

In order to power the system, a solar panel and control system had to be implemented. A typical solar power controller has six terminals. The controller that was used is shown below in Figure 130.

Figure 120 KONEZE CMP12 Solar Charge Controller.

Wiring the 6 terminals is easy to understand based on the pictures on the controller's case. The light bulb symbol represents the system load. With the previous system, the Ituner switch was used to go between direct grid power and battery power. In the solar powered system, this switch is not needed as the system is essentially always battery powered. Therefore the Ituner could be eliminated from the system, simplifying the power supply system.

An arrangement of four panels were attached and mounted to the system as shown in Figure 131. These four panels are connected in parallel. Adjustable arms where designed to allow for aiming the panels during testing. Each panel is rated at 10 Watts (40 Watt total). Although detailed calculations could be made for power budgeting, it was felt that testing the system in order to see how it would perform in the field would be quicker. A 4 panel arrangement was used initially for test which was considered the practical maximum. If the available solar power seemed to be excessive, the wiring was designed such that a panel could easily be disconnected.

Figure 121 Mounted solar panel arrangement (ALEK0® 10W Monocrystalline solar panel).

Reducing Power Consumed by the Raven Modem

The biggest challenge in creating a solar powered unit is to reduce power consumption. In this system, the biggest power draws are the Logitech camera and the Raven modem. Both of these systems are held on continuously. The camera is required to be on at all times. However, the modem is only required when communication occurs. The modem is required for communication for three purposes. One is to allow for the transmission of health checks. The second is to provide a time stamp to the impact notifications. The third is to allow access to the locally stored video data. The first two are very similar in function, while the third is very distinct and requires some additional handling. A tremendous amount of power savings can be gained by only powering the modem when needed.

The health check and impact notifications are a transmission path that come from the sensor to the user. In both cases, the Arduino controller will turn the Raven on when needed. The biggest aspect of this is that when the Raven is turned on there must be enough time between the "On" command and the "Message" to allow the modem to boot up. This means that the time stamp associated with the message will have a fixed offset to the camera timeline. As long as this offset is known, there will be no additional difficulty in locating the event in the camera's timeline interface.

The more challenging sensor interface has to do with actually downloading any specific video clips. This is because access to the data will only be permitted when the Raven is on. Therefore, it was determined to design the system to keep the Raven on for a 1 hour window during the day to allow for the user to access the system. For this process, it became clear that in order to facilitate data collection, this window needed to be consistent. This consistency needs to be maintained as the system recovers from a loss of power. In order to do this, a real time clock was added to the Arduino electronic package.

Generally speaking, limiting the time the Raven modem is on significantly reduces the power it consumes. In order to paint a clear picture of this, say that the access window is set to 1 hour (as mentioned above), modem on time for health checks is 5 minutes, and there are no impacts. From before the health checks occurred every 7 hours. Therefore, in a single 24 hour period there could be a maximum of 4 health checks. This means that the total Raven on time for a day

would be 80 minutes vs. 1,440 minutes when run continuously. Roughly speaking this means that the power consumption is reduced by 94%. This is not absolutely accurate as more power is consumed during modem startup verses when the modem is idle, but the general idea remains true. Turning the modem on and off significantly reduces the power consumed by this part of the system.

Changes Inside the Control Box

Various changes occurred inside the control box in order to implement the solar powered system and reduce the power consumption. Some minor changes were made to allow some reduction in power consumptions. The two most significant changes were the implementation of the real time clock and giving the Arduino the ability to control the modem power.

One subtle change was replacing the Waytech low-voltage disconnect element. The system allows for setting the cut out voltage of the system. This disconnect has a non-adjustable dead band between the cut out voltage and the cut in voltage. It was decided that for the solar powered system, it would be beneficial to replace this unit with a similar product that allowed for adjustability of both the cut in a cut out voltages. A Reuk Programmable low-voltage disconnect element was used. This unit was connected on the load side of the solar panel controller. The cut out voltage was set to 12.2 volts and the cut in voltage was set to 12.6. Some key system components begin to have intermittent operation when the voltage gets below 12.2 volts. The 12.6 voltage cut in voltage is used to give a minimal gap between the two voltages. This reduces the system down time for when the system is able to recharge itself. Figure 132 below gives a basic overview of the solar powered unit's power supply system.

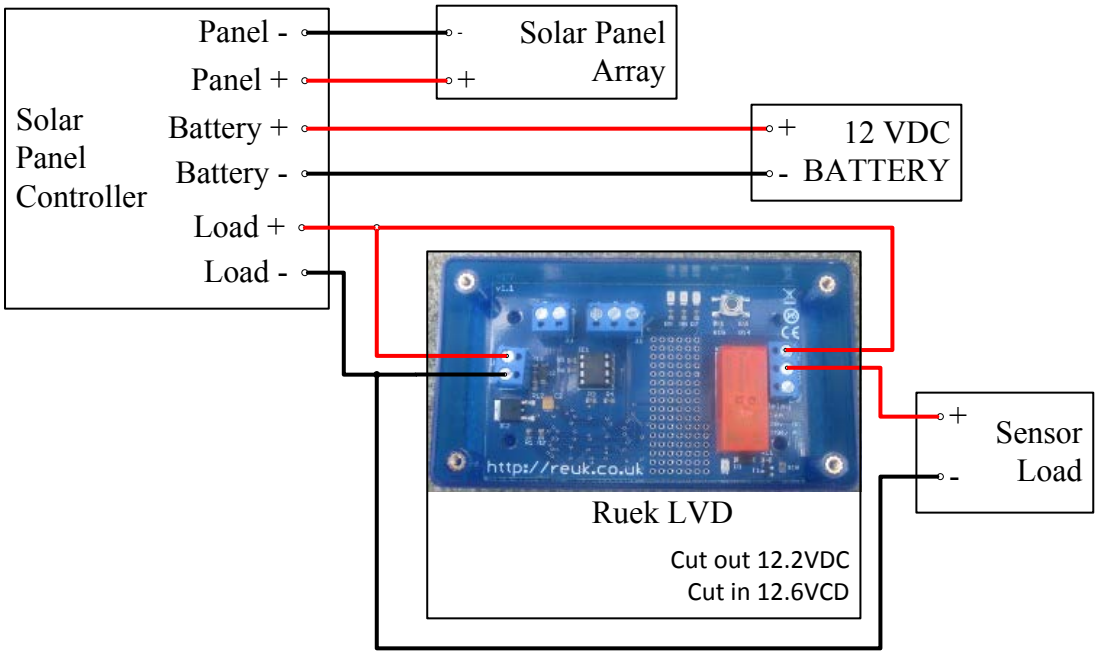

Figure 122 Updated power supply system with Ruek low voltage disconnect.

Then next major electrical change to the system is to allow the Arduino to turn the Raven on and off. This required moving the Raven's power feed connection from a terminal strip to the shield that is stacked on the Arduino board. The schematic below in Figure 133 shows the circuit that is used to control the Raven power. Implementing this circuit allows the Raven to be switched on and off with the microcontroller, thus conserving power as discussed above.

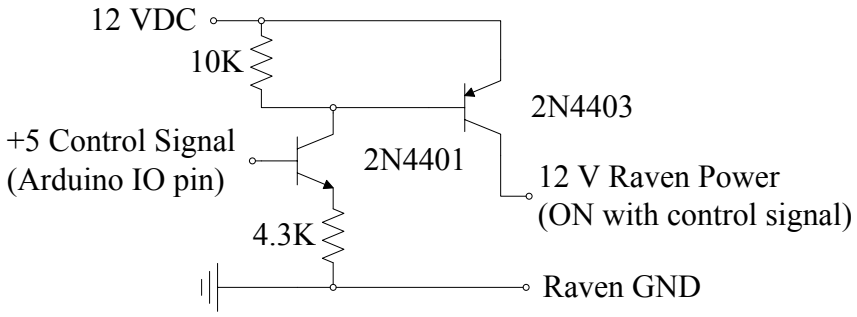

Figure 123 Raven power control circuit.

The last major electrical change was the integration of a real time clock into the system. The real time clock is required in order to allow for a consistent time window in which to access the camera data. The clock has a built in battery backup which keeps the clock running when power is turned off to the microcontroller. This allows the user to pick a specific time window for accessing the data which will be consistent even after power is lost. This window is coded in the microcontroller's firmware. The clock that was selected was a DS1307 real time clock (Adafruit part number 264). This clock uses the SPI bus on the Arduino which ties up analog pins 4 (SDA) and 5(SCL). Two other connections to clock are required which are the 5V power and ground. The basic schematic of this circuit is shown below in Figure 0.5.

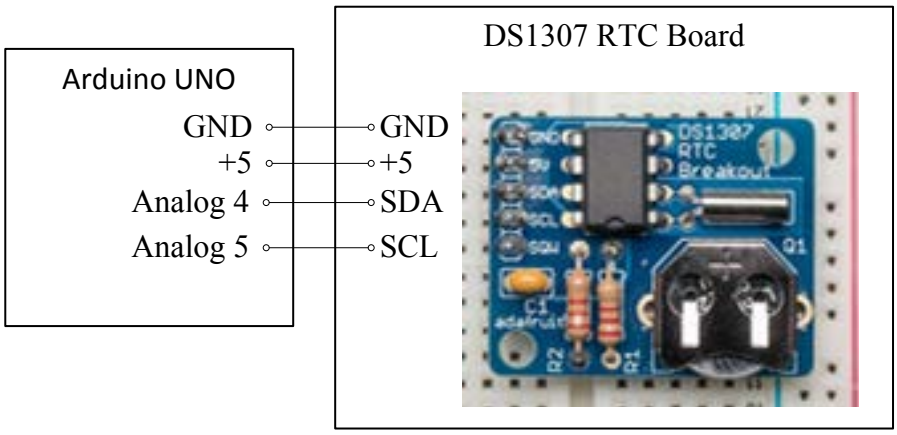

Figure 124 Real time clock circuit.

On the initial sensor development, all the Arduino's analog pins were used. The Arduino Uno has six analog IOs, and each accelerometer requires three analog pins. For the initial installation, effort was made to have two accelerometers electrically connected and program the controller to recover in the event of an accelerometer failure. This was accomplished by implementing an

error detection scheme in the control code. During the deployment, no accelerometer failures were noticed, and hence the system was rewired such that only one accelerometer is connected. In the event that an accelerometer failed, the system could be wired such that the secondary accelerometer could easily be switched out in the field.

The above changes were implemented into the test system for testing the solar powered sensor unit for it viability as a possible sensor power system. Efforts were made to reduce power consumption when possible while not causing any significant loss in usability.

System Testing

After implementing the above changes, the system was placed outside at the ATIRC facility in Davis. An additional circuit was added to the unit for testing purposes. A solar powered unit requires sunlight to operate, so time of year will play a significant role in system performance.

In order to test the system it was felt that having a mechanism to quickly look at voltages and currents in the system seemed important. The meter that was selected for this was a BeesClover 0-100V/A Digital DC Ammeter/Voltmeter. Two of these meters were implemented. One meter is used to monitor the Solar panel output to the charge controller. The other unit is used to monitor the load side of the sensor. Both of these meters were connected to a switch that turns the meters on and off. The reason is to ensure that a minimal amount of power will be used by the meters and hence affect the system power consumption during testing. This monitoring circuit is shown in Figure 135 below. The meter measures the voltage at the R2 pin. The current is measured on the negative load side through the Y lead which is internally connected to K1, which is ultimately connected to the respective systems ground.

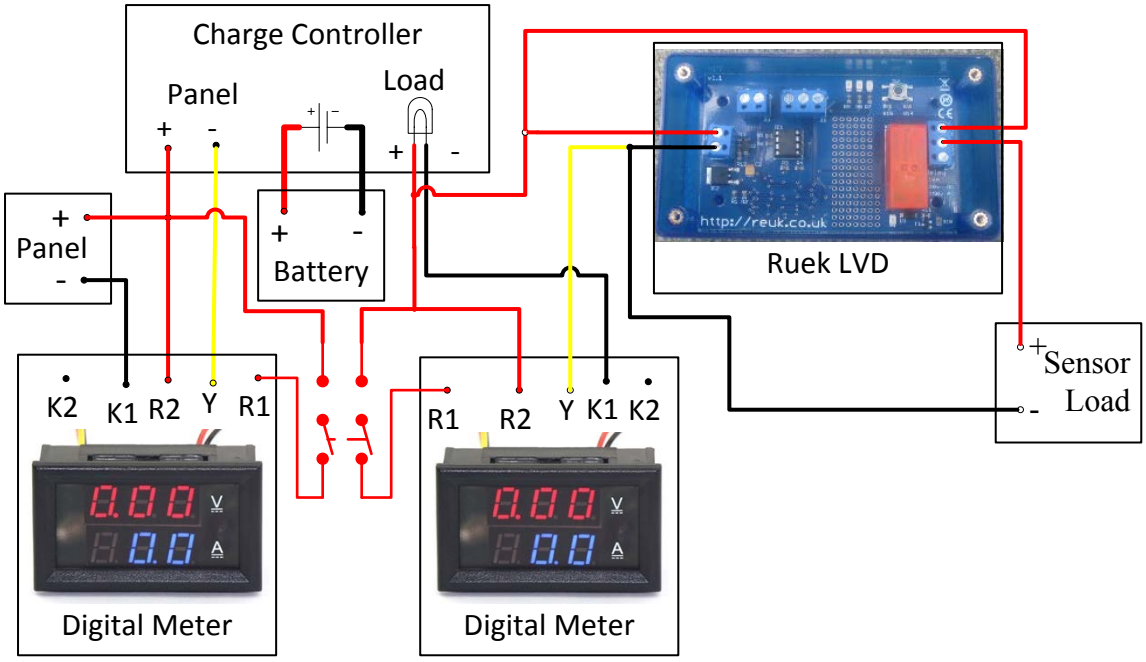

Figure 125 Power monitoring circuit.

The unit was placed outside and tested at the ATIRC facility. The testing occurred during the winter. This means that the system was tested during the time of year when the days are shorter. During testing, the highest panel voltage to the charge controller was 17.8 volts with a current output of .9 amps. The highest amount of current supplied by the panel was 1.4amps. Sensors current consumption was typically 0.31 amps. This number increased to .47 amps when the Raven was switched on. During testing, the lowest bus voltage (equivalent to battery voltage) that was detected was 12.6 volts which occurred at 6:30am. This means that the system was able to operate continuously throughout the night.

Arduino Source Code

```
// NOTES
// 1) NEED TO UNDERSTAND WHAT HAPPENS WHEN MILLIS RUNS OVER
// 2) Do I need to disable serial print is not available?
// 3) Can use button to "Double Click" for error code resetting
// 4) Can Build a more robust for serial.print ln by using string manipulation
// 5) BLINK RATE AT THE END COULD BE USED FOR BETTER USER INTERACTION
//
// Arduino Digital IO pins
int camera_led=3; //This is the light for the camera - This should be wired such that the LED is withing the camera FOV.
This pin is set to HIGH
// once an initial trip is detected.
int t1 = 9; //This is the PIN ID for IO flag pin on the Raven.
int t2 = 4; //Pin ID for send message IO flag on the Raven
// NOTE: IO's 1 and 2 of the Raven must be configured accordingly through the ALEOS software
int set_trigger_pin = 2; //This pin is used for the button which allows the system to present a serial report of it's output.
This is also the IO pin used i
//the situation when the system needs to be "reset"
//by pressing and holding this button.
// Sensor Info
char sensor_names[3] = {
 'x','y','z'}; //This is the names of the sensor axis
  // Bank 1
int sensor_pins[3]={
  0,1,2}; //Analog IN PINS for sensor
  // Bank 2
  //int sensor_pins[3]={3 4 5}; //Analop IN PINS for sensor group 2
  int sensor_flags[3]={
  0,0,0}; //This idicates which pins where tripped
int sensor_ref[3]={
  0,0,0}; //Initializes Sensor baseline variable.
int cur_sens_vals[3]={
  0,0,0};  //Initialize Sensor Values
int value_to_report[3]={
  0,0,0}; //This is the analog values which can be report
int trip = 0; //Master trip flag
// Settings
int delta=80; //This is the delta value for analog read - establishes sensativity
int loop_mon = 500; //This is the time number of iteration to loop on after the sensor is tripped
int set_trigger = 0; //Value associated with the trigger pin
int button_held = 0; //Button to telling that the system needs to be reset
//Error parameters
int n_mes_max = 4; //The maximum number of messages to send in a time span of t
int n_mes_count = 0; //The number of messages sent over the maximum time span interval
unsigned long t_vec[4]={
  0L,0L,0L,0L}; //This MUST MATCH THE NUMBER OF n_mes_max
unsigned long t_mes_max = 400000L; // millisecond interval for maximum messages
unsigned long t_cur=0L; //The current time
int num_reduce=0; //The number of integers I can reduce and shift the time vector
//Health Status - let me know your are running
unsigned long dt_Health_check = 25200000L; //1 millisecond health check interval
unsigned long Health_start = millis();  //This is the time of the last health message
unsigned long Health_cur = millis();  //The current time of the health system
int trigger_error =0; //This is the critical error flag
void setup()
{
  //Raven Trigger Pins Setup
  pinMode(t1,OUTPUT);
  pinMode(t2,OUTPUT);
  pinMode(set_trigger_pin,INPUT);
  digitalWrite(t1,LOW);
  digitalWrite(t2,LOW);
  pinMode(camera_led,OUTPUT);
  digitalWrite(camera_led,LOW);
  Serial.begin(115200);
  Serial.println("Establishing POLO Robitics Baseline Values");
  for (int idx = 0; idx <3; idx++)
  {  //Get analog sensor baseline values
```

```
    sensor_ref[idx]=analogRead(sensor_pins[idx]);
    Serial.println(String("The Nominal Value for " + String(sensor_names[idx]) + " is: " + String(sensor_ref[idx])));
    delay(10);
  } //End loop get sensor baseline values
  Serial.println("Baseline Established");
  trigger_error =0;
  Health_start = millis();
   // Serial.println("REACT Attenuator");
   Serial.println("Smart Cushion");
   //Serial.println("Compressor");
  delay(10);

}
void loop()
{
  set_trigger=digitalRead(set_trigger_pin);  //This will check to see if a button was pressed
  //NOTE: WHEN THE BUTTON IS PRESSED, THE DIGITAL INPUT GOES TO ZERO THEREFORE THE LOGIC IS
INVERTED - TRUE = NOT PRESSED
  if (set_trigger < 1)
  {  //If BUTTON PRESS = TRUE
    delay(1000);
    button_held = digitalRead(set_trigger_pin);
    if (button_held == 0)
    { //IF BUTTON HELD ==TRUE
      Serial.println("THE BUTTON WAS HELD - Reinitializing parameters");
      for (int kdx = 0; kdx <3; kdx ++)
      {//FOR LOOP BLINK LIGHT (3X)
        digitalWrite(t1,HIGH);
        delay(200);
        digitalWrite(t1,LOW);
        delay(200);
      }//END FOR LOOP BLINK LIGHT (3X)
      t_cur=millis();
      trigger_error=0;
      for (int idx = 0; idx <3; idx++)
      {  //FOR ALL ANALOG VALUES
        sensor_ref[idx]=analogRead(sensor_pins[idx]);
        Serial.println(String("The Nominal Value for " + String(sensor_names[idx]) + " is: " + String(sensor_ref[idx])));
        delay(10);
        sensor_flags[idx]=0;
        cur_sens_vals[idx]=0;
        value_to_report[idx]=0;
      } //END FOR ALL ANALOG VALUES
      Serial.println("Baseline Established");
      trigger_error =0;
    //  Health_start = millis();
      n_mes_count = 0;
      t_vec[0]=0L;
      t_vec[1]=0L;
      t_vec[2]=0L;
      t_vec[3]=0L;
      num_reduce=0;
      trip = 0; //Master trip flag
    } //END IF BUTTON HELD == TRUE
    else if (button_held ==1)
    {//ELSE IF BUTTON HELD == FALSE
      Feedbackbutton_FCN(loop_mon, delta,
     sensor_ref,sensor_names,dt_Health_check,Health_start,Health_cur,t_mes_max,t_cur,n_mes_max);
    }//EMD ELSE IF BUTTON HELD == FALSE
  }  //END IF BUTTON PRESS = TRUE
  delay(20);
  Health_cur = millis();
  if (Health_cur < Health_start)
  { //IF-ELSE HEALTH CONTROL - THE MILLIS COUNTER OVERFLOWED = true
    //Dealing with arduino overflow in millis - according to the web. should be about 50 day interval
    //Strategy is that the Health_start will only be greater then the current time on "Rollover" Day
    //This does not protect from issues if the time between impact "1" occurs < 10 minutes prior to rollover, and then
    Health_start=Health_cur;
    Serial.println("Resetting Health Start");
  } //END IF-ELSE -  THE MILLIS COUNTER OVERFLOW = TRUE
```

```
 else if (Health_cur - Health_start > dt_Health_check)
 {//IF-ELSE HEALTH CONTROL - TIME TO SEND HEALTH STATUS UPDATE
  //Begin Health Check feedback
  Health_start=Health_cur;
  Serial.println(("Health Status: " + String(trigger_error) + "\n 0 = Unhealthy \n 1 = Healthy"));
  //Need to trigger the messaging - Single message means startup
  SendMessage_FCN(trigger_error,t1,t2);
  //The Camera will send a health check message and blink the led to ensure things are working.
  for (int idx = 1; idx <15; idx++)  //FOR BLINK CAMERA LED
  {
   digitalWrite(camera_led,LOW);
   delay(80);
   digitalWrite(camera_led,HIGH);
   delay(80);
 } //END FOR BLINK CAMERA LED
 digitalWrite(camera_led,LOW);
 }    //END IF-ELSE HEALTH CONTROL - TIME TO SEND HEALTH STATUS UPDATE
 if (trigger_error ==1)
 { // IF ELSE TRIGGERING ERROR = TRUE  - locks the system in this mode so that it will continually run and not send
messages
  Serial.println("CRITICAL ERROR");
  digitalWrite(t1,HIGH);
  delay(500);
  digitalWrite(t1,LOW);
  delay(500);
  //Could send a message Here
 } //END IF-ELSE TRIGGERING ERROR = TRUE
 else
 { //IF-ELSE TRIGGERING ERROR = FALSE
 //This is the normal operational loop of the code
  for (int idx =0; idx<3; idx++)
  { //FOR ALL SENSORS
   cur_sens_vals[idx]=analogRead(sensor_pins[idx]);
   if (cur_sens_vals[idx] >sensor_ref[idx]+delta)
   { // IF POSITIVE INITIAL TRIGGER = TRUE
    digitalWrite(camera_led,HIGH);
    trip =1;
    digitalWrite(camera_led,HIGH);
    sensor_flags[idx]=1;
    value_to_report[idx]=cur_sens_vals[idx];
    Serial.println("Sensor Was Tripped (+accel.)");
    Serial.println(sensor_names[idx]);
    break; //BREAK END FOR ALL SENSORS
   }  // END IF POSITIVE INITIAL TRIGGER = TRUE
   if (cur_sens_vals[idx]<sensor_ref[idx]-delta)
   {  //IF NEGATIVE INITIAL TRIGGER = TRUE
    trip = 1;
    digitalWrite(camera_led,HIGH);
    sensor_flags[idx]=1;
    Serial.println("Sensor Was Tripped (-accel.)");
    value_to_report[idx]=cur_sens_vals[idx];
    Serial.println(sensor_names[idx]);
    break; // BREAK END FOR ALL SENSORS
   }// END IF NEGATIVE INITIAL TRIGGER = TRUE
  }// END FOR ALL SENSORS
  if (trip > 0)
  { //IF INITIAL TRIP WAS DETECTED
   t_cur=millis();
   for (int l_count=0; l_count<loop_mon; l_count ++)
   { //FOR LOOP MONITORING - This is intended to give a dt so that are sensors
    //can be check for a period of time to enable loop limitations
    for (int idx =0; idx<3; idx++)
    {//FOR ALL ANALOG PINS
     cur_sens_vals[idx]=analogRead(sensor_pins[idx]);
     if (cur_sens_vals[idx] >sensor_ref[idx]+delta)
     { // IF POSITIVE TRIGGER = TRUE
      sensor_flags[idx]=1;
      value_to_report[idx]=cur_sens_vals[idx];
     }  // END IF POSITIVE POSITIVE TRIGGER = TRUE
     if (cur_sens_vals[idx]<sensor_ref[idx]-delta)
```

```
        {  //IF NEGATIVE TRIGGER = TRUE
          sensor_flags[idx]=1;
          value_to_report[idx]=cur_sens_vals[idx];
        }// END IF NEGATIVE TRIGGER NEGATIVE TRIGGER = TRUE
      }// END FOR ALL ANALOG PINS
    } // END FOR LOOP MONITORING
     for (int idx = 1; idx <15; idx++)  //FOR BLINK CAMERA LED
   {
    digitalWrite(camera_led,LOW);
    delay(80);
    digitalWrite(camera_led,HIGH);
    delay(80);
  } //END FOR BLINK CAMERA LED
  digitalWrite(camera_led,LOW);
    // Sensor Was Tripped NEED TO WORK ON WRITING OUT INFORMATION
    Serial.println("Begin Message");
    for (int idx = 0; idx <3; idx++)
    { //FOR ALL PINS
     if (sensor_flags[idx] ==1)
     {//IF SENSOR[idx] = TRUE
       Serial.println(sensor_names[idx]);
       Serial.println(value_to_report[idx]);
       Serial.println("Was Tripped");
     } // END IF SENSOR[idx] = TRUE
     if (sensor_flags[idx] < 1)
     {// IF SENSOR[idX] = FALSE
       Serial.println(sensor_names[idx]);
       Serial.println("Was NOT Tripped");
     }// END IF SENSOR[idx] = FALSE
     SendMessage_FCN(sensor_flags[idx],t1,t2);
    } // END FOR ALL PINS
    // DONE WITH MESSAGING - NEED TO RESET VALUES
    trip = 0;
    digitalWrite(camera_led,LOW);
    for (int idx = 0; idx <3; idx++)
    { //FOR CLEAR VALS FOR ALL PINS
     sensor_flags[idx]=0;
     value_to_report[idx]=0;
    }// END FOR CLEAR VALS FOR ALL PINS
    Serial.println("Message Complete");
    Serial.println("Waiting to Establish New Baseline");
    delay(4000);
    for (int idx =0;idx<3; idx++)
    {//FOR ALL PINS - REDEFINE BASE VALUES
     sensor_ref[idx]=analogRead(sensor_pins[idx]);
    }//END FOR ALL PINS REDEFINE BASE VALUES
    Serial.println("Done Establishing New Baseline");
    // LOOK AT TIME ISSUE FOR FAILUES
    n_mes_count=n_mes_count+1; //This is the number of messages that has been sent
    num_reduce=0;
    if (t_cur-t_vec[0] < 0)
    {//IF THE MILLIS NUMBER OVERFLOWS = TRUE
     //Will reset all values - low probability and will cause a few extra messages to be allowed in the right condition.
     n_mes_count = 1;
     t_vec[0]=0L;
     t_vec[1]=0L;
     t_vec[2]=0L;
     t_vec[3]=0L;
    }//END IF THE MILLIS NUMBER OFVERLOW = TRUE
    for (int kdx=0; kdx<=n_mes_count-2; kdx++)  //Find out how much I can reduce the time vector by
    { //FOR ALL COUNTED IMPACTS
     if ((t_cur - t_vec[kdx] ) > t_mes_max)
                { //IF TIME BETWEEN CUTTENT IMPACT AND OTHER IMPACT EXCEEDS THE LIMIT CAN
          ELIMINATE = TRUE
       num_reduce=num_reduce+1;
                } //END IF BETWEEN CUTTENT IMPACT AND OTHER IMPACT EXCEEDS THE LIMIT CAN
          ELIMINATE = TRUE
    }//END FOR ALL COUNTED IMPACTS
    if (n_mes_count > n_mes_max && num_reduce ==0)
    {//IF-ELSE ERROR ENTRY CONDITION ERROR = TRUE
```

```
      //Exceeded the number of messages in time
      Serial.println("Too Many Triggers - Critical Error - Must restart controller");
      trigger_error =1;
      // COULD RE-EVALUE t_vec here, but it is done when the user resets the system
      //       t_vec[0]=0L;
      //       t_vec[1]=0L;
      //       t_vec[2]=0L;
      //       t_vec[3]=0L;
     } //END IF-ELSE ERROR ENTRY CONDITION =TRUE
     else
     {//IF-ELSE ERROR ENTRY CONDITION ERROR = FALSE
      if (n_mes_count==1) //IF-ELSE FIRST MESSAGE = TRUE
      {//IF-ELSE FIRST MESSAGE = TRUE
       t_vec[0]=t_cur;
      }// END IF-ELSE FIRST MESSAGE == TRUE
      else
      {  //IF-ELSE FIRST MESSAGE == FALSE
       for (int jdx=1; jdx<=(n_mes_count-num_reduce-1)+1; jdx++)
       { //FOR ALL TIME VALUES THAT MUST BE KEPT
        t_vec[jdx-1]=t_vec[num_reduce+jdx-1];
       } //END FOR ALL TIME VALUES THAT MUST BE KEPT
       n_mes_count=n_mes_count-num_reduce;
       t_vec[n_mes_count-1]=t_cur;
       for (int kdx=n_mes_count+1; kdx<=n_mes_max; kdx++) //SET_Z_VALLUES
       { //FOR ALL TIME VALUES THAT REMAIN - SET VALUES TO ZERO
        t_vec[kdx-1]=0L;
       } //END FOR ALL TIME VALUES THAT REMAIN
      }//END IF-ELSE FIRST MESSAGE == fALSE
     }//END IF-ELSE ERROR ENTRY CONDITION = FALSE
     Serial.println(t_vec[0]);
     Serial.println(t_vec[1]);
     Serial.println(t_vec[2]);
     Serial.println(t_vec[3]);
     delay(200);
     Serial.println("Waiting for input");
    }
   }//END TRIP WAS DETECTED
}
//This part of the code allows the user to get current status information from the board through the Serial communication
interface
void Feedbackbutton_FCN(int loop_mon, int delta, int sensor_ref[3], char sensor_names[3],unsigned long
dt_Health_check, unsigned long Health_start, unsigned long Health_cur, unsigned long t_mes_max, unsigned long t0, int
n_mes_max){
  //Some issues here due to text formaatting which should be resolved at some point.
  Serial.println("BEGIN SOFTWARE FEEDBACK \nThe IO button is to allow for serial notification of the current
settings");
  //  delay(10);
  // Serial.println("The IO button is to allow for serial notification of the current settings");
  delay(10);
  Serial.println("This code was written on December 10,2012");
  delay(10);
 // Serial.println("REACT Attenuator");
    Serial.println("Smart Cushion");
   //  Serial.println("Compressor");
  delay(10);
  Serial.println(String ("The loop monitoring time span is: " + String(loop_mon) + " - This signifies how long to look at the
accelerometer prior to sending a message"));
  delay(10);
  Serial.println(String("The system relies on a +/-" + String(delta) + " deviation from the nominal value for triggering"));
  for (int idx=0; idx <3; idx++)
  {
   Serial.println(String("The Nominal Value for " + String(sensor_names[idx]) + " is: " + String(sensor_ref[idx])));
   delay(10);
  }
  Serial.println(String("The utility has a critical error when more then " + String(n_mes_max) + " messages are sent in less
then " + String(t_mes_max) + " ms"));
  delay(10);
  Serial.println(String("Currently only : " + String(n_mes_count) + "messages  have been sent starting a t =" + String(t0)));
  delay(10);
```

```
  Serial.println(String("This system will send health check messages every " + String(dt_Health_check) + "ms.  The last
one was sent at: " + String(Health_start) + "ms"));
  delay(10);
  Serial.println(String("The Current clock time is:" + String(Health_cur) + "ms"));
  delay(10);
  Serial.println("END SOFTWARE FEEDBACK");
  delay(10) ;
  Serial.println("Waiting before resume");
  delay(2000);
  Serial.println("Running Normal");
  return;
}
void SendMessage_FCN(int FLAG,int FLAG_PIN, int MSG_PIN) {
  //Flag Will be a 1 or a zero depending on contact
  //FLAG_PIN - IO to trigger for message
  //MSG_PIN - Pin TO use to send message
  if (FLAG ==1) {//This sensor was tripped generator output
    digitalWrite(FLAG_PIN,HIGH);
  }
  delay(500);
  digitalWrite(MSG_PIN,HIGH);
  delay(500) ;
  digitalWrite(MSG_PIN,LOW);
  delay(500);
  digitalWrite(FLAG_PIN,LOW);
  delay(500);
```