STATE OF CALIFORNIA • DEPARTMENT OF TRANSPORTATION

TECHNICAL REPORT DOCUMENTATION PAGE

TR0003 (REV 10/98)

ADA Notice

For individuals with sensory disabilities, this document is available in alternate formats. For information call (916) 654-6410 or TDD (916) 654-3880 or write Records and Forms Management, 1120 N Street, MS-89, Sacramento, CA 95814.

1. REPORT NUMBER	2. GOVERNMENT ASSOCIATION NUMBER	3. RECIPIENT'S CATALOG NUMBER
TR-IEL-2003-103		

4. TITLE AND SUBTITLE

Non-Intrusive Methods of Characterizing Vehicles on the Highway

5. REPORT DATE

June 30, 2003

6. PERFORMING ORGANIZATION CODE

7. AUTHOR

Harry H. Cheng, Ben Shaw, Joe Palen, Zhaoqing Wang, Bo Chen, Steve Nestinger

8. PERFORMING ORGANIZATION REPORT NO.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Integration Engineering Laboratory
Department of Mechanical and Aeronautical Engineering
University of California, Davis
Davis, CA 95616

10. WORK UNIT NUMBER

11. CONTRACT OR GRANT NUMBER

65A0121 A01

12. SPONSORING AGENCY AND ADDRESS

California Department of Transportation
Sacramento, CA 95819

13. TYPE OF REPORT AND PERIOD COVERED

Final June 2001 - June 2002

14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

16. ABSTRACT

Over the past year we have worked on the development of a real-time laser-based non-intrusive field-deployable detection system for delineation of moving vehicles. The primary goal of the project is to develop a roadway detection system that can be used to gather reliable travel time data non-intrusively. Microprocessor is used to control digital-controlled potentiometer (DCP), which adjusts the gain of sensors' signals. It adjusts the system quickly on highway with only pushing a button. The adjustment, which used to take half or an hour, needs only several seconds. Meantime, the microprocessor is also able to filter noise after having finished the task of adjustment. The software of the system is rebuilt for 8 channels, which used to be for 4 channels, to be able to obtain more information of vehicles even like the outline of vehicle. In order to improve the precision of the system, we improved the mechanical design, optical design and electric circuit design. The frequency of laser pulse is 10 kHz and the sample rate of computer DIO96 board is also 10 kHz. This document describes the design and implementation of each functional component of the field-deployment system, the configuration of the field detection system, and software design and implementation. This report also discusses optical techniques that may potentially be used for the detection of trace amounts of gases such as CO, CO2, NO and NO2 through the use of an overhead detector that employs signal reflection from a roadway like the Laser-Based Detection System. These gas species are relevant to detection of highly polluting vehicles.

17. KEY WORDS	18. DISTRIBUTION STATEMENT	
Traffic, Caltrans; Real Time, Laser, Safety, Electronics, Loop, Detector, Gas, Speed		
19. SECURITY CLASSIFICATION *(of this report)*	**20. NUMBER OF PAGES**	**21. COST OF REPORT CHARGED**
unclassified	113	

Technical Report TR-IEL-2003-103

Non-Intrusive Methods of Characterizing Vehicles on the Highway

Harry H. Cheng
Ben Shaw
Joe Palen
Zhaoqing Wang
Bo Chen
Steve Nestinger

Integration Engineering Laboratory
Department of Mechanical and Aeronautical Engineering
University of California, Davis
Davis, CA 95616

June 30, 2003

A Project Report for California Department of Transportation
Under Contract No: 65A0121 A01

Contents

List of Figures

Abstract

Over the past year we have worked on the development of a real-time laser-based non-intrusive field-deployable detection system for delineation of moving vehicles. The primary goal of the project is to develop a roadway detection system that can be used to gather reliable travel time data non-intrusively. Microprocessor is used to control digital-controlled potentiometer (DCP), which adjusts the gain of sensors' signals. It adjusts the system quickly on highway with only pushing a button. The adjustment, which used to take half or an hour, needs only several seconds. Meantime, the microprocessor is also able to filter noise after having finished the task of adjustment. The software of the system is rebuilt for 8 channels, which used to be for 4 channels, to be able to obtain more information of vehicles even like the outline of vehicle. In order to improve the precision of the system, we improved the mechanical design, optical design and electric circuit design. The frequency of laser pulse is 10 kHz and the sample rate of computer DIO96 board is also 10 kHz. This document describes the design and implementation of each functional component of the field-deployment system, the configuration of the field detection system, and software design and implementation. This report also discusses optical techniques that may potentially be used for the detection of trace amounts of gases such as CO, CO_2, NO and NO_2 through the use of an overhead detector that employs signal reflection from a roadway like the Laser-Based Detection System. These gas species are relevant to detection of highly polluting vehicles. This report also discusses measurements of vehicle paint reflectance characteristics. These measurements were made using two portable spectrometer systems available at UC Davis. Both spectrometers employed a fiber-optic reflection probe coupled with an incandescent light source. Reflectance measurements were made from about 800 nm to about 1100 nm, a spectral range that should be invisible to the human eye. Reflectance data were taken normal to vehicle surfaces, providing specular reflection measurements, as well as at a 45-degree angle, providing diffuse reflection measurements. The data indicate that diffuse reflection characteristics show the strongest wavelength dependence and that overall IR vehicle paint reflectance levels can vary significantly with the type of paint (i.e., visible color). For some paints, IR reflectance levels sometimes varied only by small amounts (a few percent or less) over the range 850 nm to about 1000 nm, that is, vehicle paint reflectance did not vary strongly with wavelength over this portion of the spectrum. For other paints, though, larger changes were apparent. As a result, it is possible that a laser-based measurement system that measures reflectance ratios of two different wavelength lasers in the IR portion of the spectrum could be successfully developed. Because reflectance characteristics can vary somewhat slowly with wavelength, a key to the successful development of this type of system will be to employ lasers that have significantly different wavelengths.

Introduction

According to the conclusion that both CalTrans and the US Department of Transportation have come to, it is impossible to build our way out of traffic congestion. The solution is to run the transportation system more intelligently, which is known as the Intelligent Transportation System. Travel time is a good indicator of other direct constraints on ITS efficiency: cost, risk, and attentive workload. The importance of travel time is verified in Advanced Traffic Information System (ATIS) user surveys which indicate that what travelers most want from a transportation system is (almost always) reduced travel time and higher reliability (e.g. reduced travel time variance and reduced risk)[1]. Every traveler must implicitly or explicitly make an assessment of these various travel time options before embarking on every trip; therefore, this information is definitely of high value. Because trip travel time is the parameter the public most wants to minimize, this is the parameter that is most important for transportation service providers to measure and minimize.

Speed is commonly used as an indicator of the travel time across a link. In current practice, speed is measured at one or more points along a link and extrapolated across the rest of the link [1]. This extrapolation method is used regardless of the mechanism of detection. Example detection methods are loops---which determine speed from two elements twenty feet apart; radar---which can directly determine speed from the carrier frequency shift (Doppler Effect); or video image processing---which tracks vehicles across the pixel elements within the field of view. The extrapolation from a point to a line is not necessarily valid. At the onset of flow breakdown, the speed variations along the length of a link can be quite large. Also, the onset of flow breakdown is when routing decisions are most time-critical and accurate information has the highest value, so inaccurate extrapolations could have detrimental effects to the traveler.

An alternate method to determine the traverse travel time (e.g. the true link speed) is to use Vehicles as Probes (VAP). A VAP system determines travel time directly by identifying vehicles at the start of the link and re-identifying them at the end of the link, with the time difference being the true travel time. The problem with VAP systems is that they require large numbers of both vehicle tags and tag readers to be effective, and the cost justification of such a system seems unwarranted in the light of other options. The key aspect to measuring the actual travel time is simply to identify some distinguishing characteristic on a vehicle at the beginning of a link and then to re-identify that same characteristic on the same vehicle at the end on the link. This is the basic idea of VAP, however the characteristic does not have to be entirely unique (as in a vehicle tag), and it does not necessitate the infrastructure set-up costs of VAP. If a characteristic can be found to separate the fleet into (say) 100 classifications, ``the maximum probability fit'' can be determined for the same sequence of classifications at the downstream detector as was identified at the upstream detector. This is what is currently being done in Germany with the low-resolution imaging provided by (new high speed) loops [1]. If a higher-resolution detector is used so that it is possible to get a few thousand classes, then it should be quite possible to perform a 100% upstream-downstream Origin and Destination (O/D) analysis (even if a significant percentage of the vehicles switch lanes) using time gating and other relatively straight-forward signal processing techniques. The mechanism of detection must allow highly resolved delineations between commonly available “commuter” vehicles, because commuter vehicles represent the majority of the vehicle stream during the period that travel time information is most needed (e.g. the peak hours).

Any mechanism to measure travel time, by definition, is only determining the “past state” of the transportation system. Collecting data on what happened in the past has no utility except if it is used to infer what may happen in the future. All decisions, by definition, are based on an inference of future consequences. When a traveler learns that speed on a route is 50 MPH, the traveler generally infers that the speed will remain 50 MPH when she/he traverses it. This may or may not be an accurate inference. Travelers want to know the “state” of the system (in the future) when they traverse it. In the simplest case, this is just a straight extrapolation of current “state”. More sophisticated travelers may develop their own internal conceptual model of the typical build up and

progression of congestion along routes with which they are familiar. A major benefit of ITS will be to provide travelers with a much more valid and comprehensive "look ahead" model of the (short-term) future state of the transportation system. Validation of any traffic model requires (either implicitly or explicitly) traffic O/D data. The lack of valid O/D data has been the major impediment in the calibration, validation and usage of traffic models. In this research project we are developing a roadway detection system that can directly determine O/D data non-intrusively without violating the public's privacy (as in license plate recognition systems).

Having been studied on the project of Real-Time Laser-Based Non-Intrusive Detection System for Measurement of true Travel Time on Highway, we improved the system further more. A Feed-Back Loop, using Microchip to control, was introduced into the system. It makes the system easy to use. The mechanical was improved to high the precision of the system. Software to display the information or communicated with other research group was updated. The new system device is showed in Figure 0-1 below.

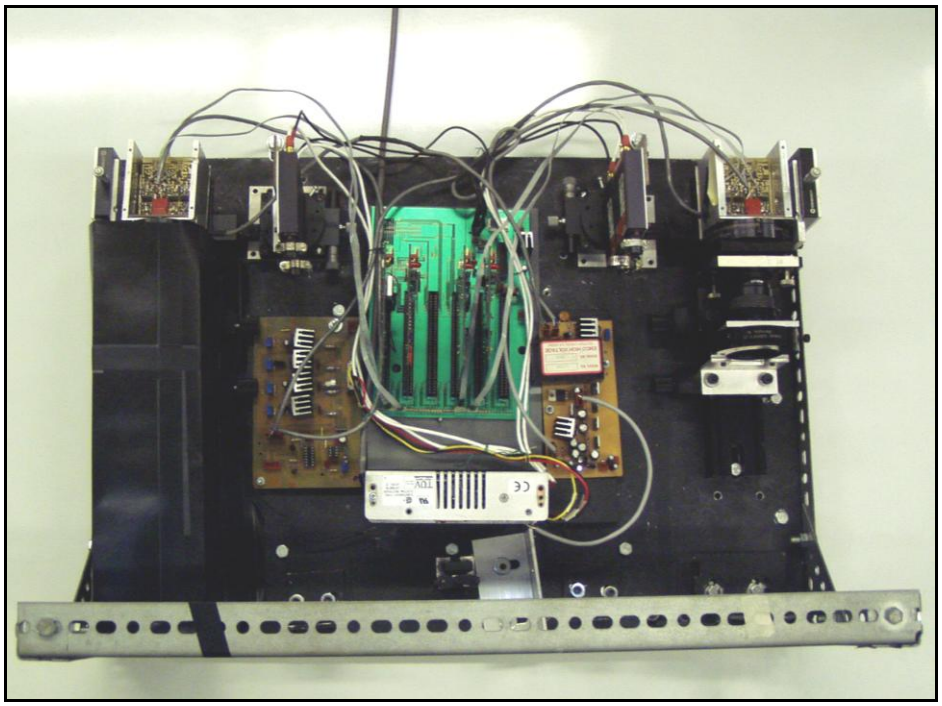


Figure 0-1: Current Prototype of the Laser-Based Detection System

1 Overview of the Laser-Based Detection System (LBDS) Project

Figure 1-1, below, shows a conceptual drawing of the LBDS. It is comprised of two laser diode units and other optical, mechanical, electronic components. The basic detector unit consists of a laser diode system and a photodiode array positioned above the roadway. The laser system is a pulsed infrared diode with line-generating optics that projects a laser beam onto a road surface. The imaging lens focuses the reflected laser light onto the active area of the sensor array. The signal from a photodiode is amplified and sent to a computer for processing. Vehicle presence is detected based on the absence of the reflected laser light. Two detector units are integrated and placed at a known distance apart. Two pairs of lasers/return optics are used for detecting vehicle presence at two points laterally about 4 and 8 feet across the lane. A sampling rate of about 10 kHz is generally acceptable. Any (threshold) change in the amplitude of the pavement reflectance indicates a vehicle. As shown in Figure 1-1, t_1 is the time when the first laser line is blocked, t_2 is when the second laser line is blocked, t_1' is when the first laser is unblocked, and t_2' is when the second laser line is unblock. We can calculate the length and the speed of a vehicle with D_{ref}, which is the distance between the two laser lines, t_1, t_2, t_1', and t_2'. Both signals outputted from the

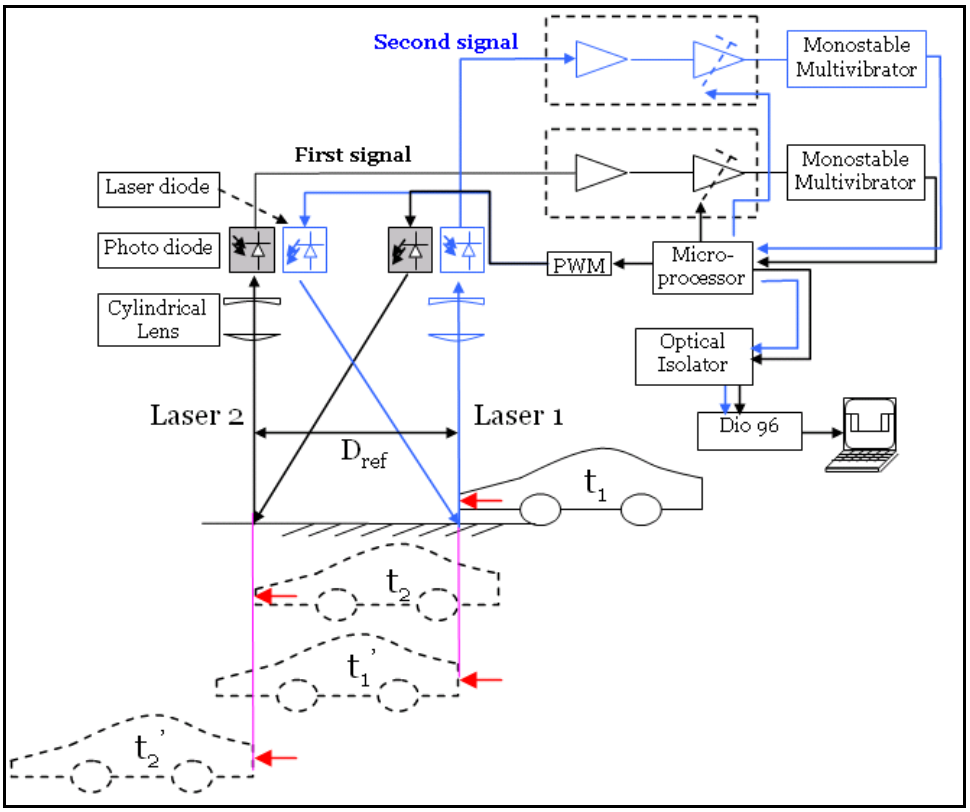


Figure 1-1: Laser-Based Detection System Overview

1.1 Integrated Laser and Optics

Four off-the-shelf integrated high power diode lasers from Power Technology Inc (model ML20A15-L2) are used in the current prototype. This is an integrated laser system that incorporates a DC/DC voltage converter, a voltage regulator, a pulse generator, a laser diode, and line generation optics all into one unit. The unit only requires a 12V DC power supply and a trigger pulse for regular

operation. Each laser has a peak power output of 20W at 905nm, with a pulse width of 9ns. They can be pulsed up to a maximum rate of 10 kHz. The line generating optics produces a beam having a full fan angle of 15 degrees. The lasers' high performance and compact size make it a good candidate for use in a field deployable prototype system.

A laser producing infrared light (with a 905nm wavelength) was chosen for a number of reasons. Infrared light has good transmittance through fog, which facilitates better performance under a larger range of weather conditions. Furthermore, the intensity of sunlight near that wavelength is at a local minimum, minimizing any noise due to sunlight interference. An infrared laser was also thought to be more appropriate for outdoor use because it is invisible to the human eye, causing no distraction to passing motorists.

The laser has to be properly adjusted to obtain a fine line. The procedure for adjusting is as follows:
Adjust the focus without the cylindrical lens in the line generator.
Put the cylindrical lens on and rotate it to minimize the width of the line.

In order to retain focus, a clamp is used to tighten down the line generating optics of the lasers.

The sensor optics consists of an imaging lens system and a telescopic lens system. The imaging lens system focuses the reflected laser light onto the active area of the sensor array. The imaging lens was selected based on several criteria. It should have an adjustable focal length within a range around the desired focal length, a field-of-view large enough to capture the width of an entire lane, and be compact for easy integration into the outdoor system.

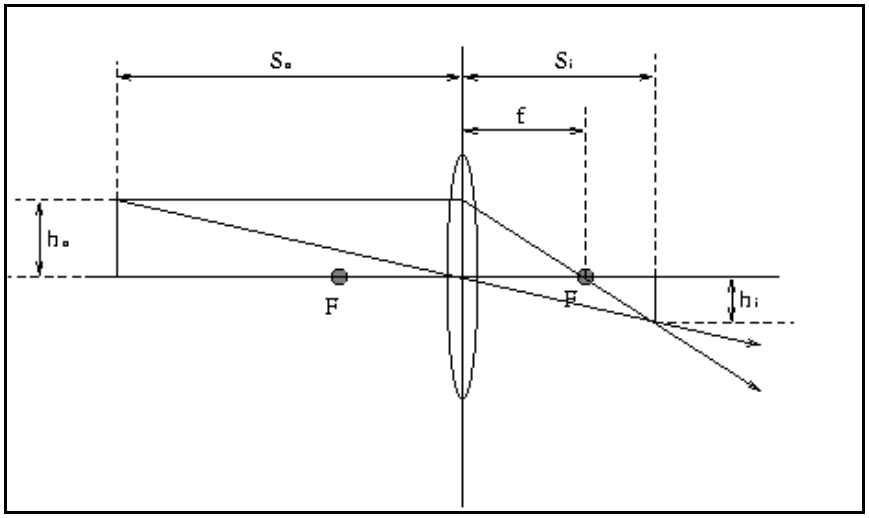


Figure 1-2: Standard Lens Parameters

Figure 1-2, above, shows standard lens parameters used in the following calculations. Based on the assumptions that the lane width (h_o) is around 3.05 m (10.0 ft), the unit will be mounted about 6.40 m (21.0 ft) above the roadway s_o ,and the sensor is 7.5 mm (0.295 in) long h_i , an image distance s_i was calculated for the sensor using Equation (1-1). It was determined that s_i = 15.8 mm (0.620 in).

$$s_i = s_o \frac{h_i}{h_o} \qquad \textbf{(1-1)}$$

The desired focal length f of the lens was then calculated using Equation (1-2).

$$f = \frac{1}{\frac{1}{s_i} + \frac{1}{s_o}} \qquad \textbf{(1-2)}$$

The focal length was calculated to be 15.7 mm (0.616 in). As a practical matter, the sensor array is placed at the focal point of the imaging lens system. Since s_o is large in comparison to s_i, f is nearly equal to s_i.

The imaging lens selected, a Tamron 23VM816, has an adjustable focal length of between 0.315 in (8 mm) and 0.630 in (16 mm) and was selected because of this feature. The Tamron lens is also suitably compact and has a field of view that should be large enough to capture the entire lane width. Any lens system that has the correct focal length and an acceptable field-of-view could be used.

The telescopic lens system is mounted in front of the imaging lens system. It is designed to restrict the field-of-view of the imaging lens along the width of the laser line, but not alter the field-of-view along the length of the line. Because the laser line is much longer than it is wide, use of the imaging lens alone would result in a much wider strip of pavement being visible to the sensor than is desired. The telescopic lens system is used to match the dimensions of the laser line image with those of the sensor array. Figure 1-2, below, shows the imaging lens and the cylindrical lenses used in our system.

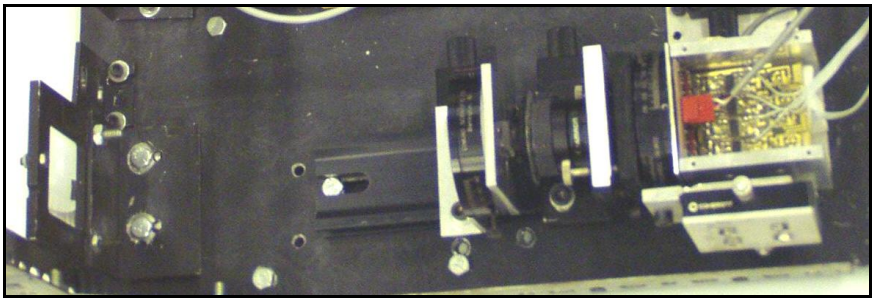


Figure 1-2: View of the Telescopic and Imaging Lenses

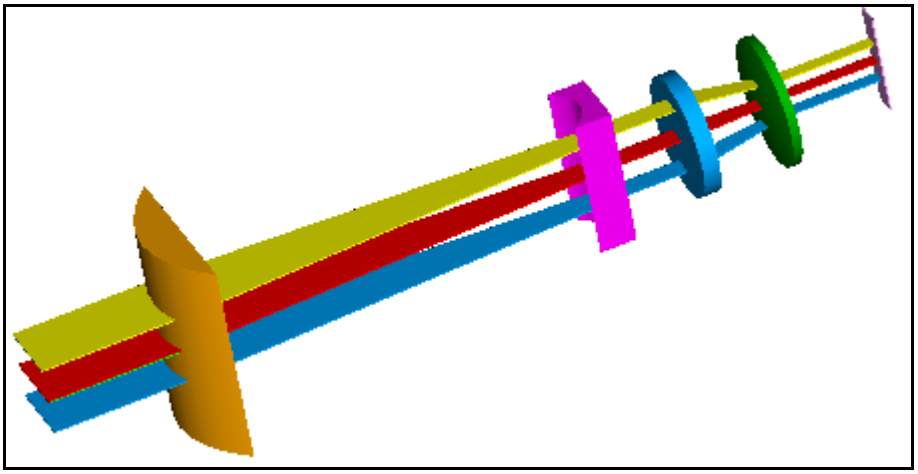


Figure 1-3: Computer View of the Optical System

The telescopic lens system consists of one positive plano-cylindrical lens and one negative plano-cylindrical lens. The prototype uses a 250 mm focal length cylindrical lens and a -12.70 mm focal

length cylindrical lens, both manufactured by Melles Griot Inc. These lenses are positioned to form a Galilean telescope. When positioned correctly the cylindrical lenses will not effect the proper operation of the imaging lens. The ratio of the focal length of these lenses is approximately equal to the ratio of the width of the uncorrected field-of-view of the sensor to the desired field-of view. The desired field-of-view X is determined from Equation (1-3), where Y is the separation of the sensor and laser, H is the height of the system above the road, and H_c is the desired minimum detectable object height. To insure reliable vehicle detection it is important that H_c be below the bumper height of most common vehicles.

$$Y = \frac{X(H - H_c)}{H_c} \qquad (1\text{-}3)$$

The uncorrected field of view, about 13cm (5 in), results in a critical height of about 1.8 m (6.0 ft). To ensure vehicle detection, it is necessary to have a critical height somewhere below the bumper height of the vehicle. A height of around 46cm (18 in) was thought to be acceptable. To achieve this, it is necessary to restrict the field of view X to about 2.3cm (0.92 in). This is a factor of reduction of about 5. In our case, where f1 = 250 mm and f2 = -12.70 mm, the factor of reduction is equal to about -7.9 (the negative sign indicates an inverted image), giving us a field of view of about 16 mm (0.63 in). The factor of reduction is commonly referred to as the angular magnification of the system. A ray of light entering the system from the left at an angle Θ1 exits the system at the right at an angle Θ2 equal to Θ1*(f1/f2). This causes objects to the left to appear larger than they actually are. This is how the field of view is reduced. A sensor on the right of the telescopic system will have its field of view reduced by a factor equal to the angular magnification of the system. The telescopic system does not alter the position or focus of the image. Objects that are properly focused by the imaging lens remain in focus when the telescopic system is added.

The laser specifications and optical specifications required for the LBDS are located under Appendix B.

1.2 Signal analysis

The basic working principle of our LBDS is shown below in Figure 1-4: the laser is projected onto the road and then reflected from the road into the receiver area. Using optical lens, the reflected laser can be focused onto a photodiode. When a vehicle passes under the system, the reflected laser is blocked and the photodiode will not generate a signal. Otherwise, the system is able to pick up the signal from the photodiode. In this system, the power of reflected laser determines the strength of the photodiode signal and ultimately affects the result of the system.

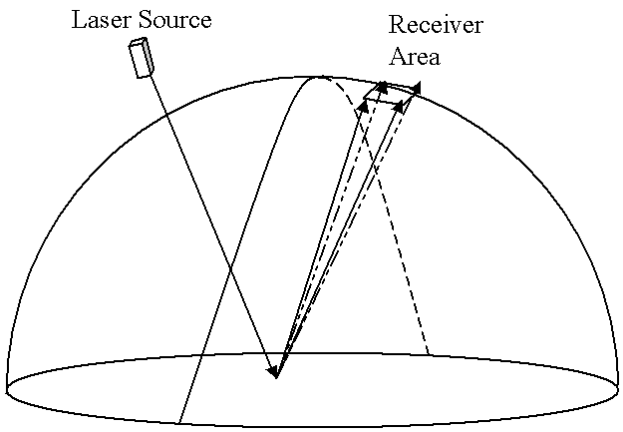


Figure 1-4: Laser Reflection

The power of reflected signal from the road is calculated by [30]:

$$P_L'' \approx \rho \frac{A_R}{2\pi R^2} P_L' \times \tau_L \times \tau_F$$

where P_L' is Laser output power,
ρ is Roadbed reflection,
A_R is the receiver lens Area.
τ_L is lens transmissivity = 0.9
τ_F is interface filter transmissivity = 0.5
In our system:
ρ=0.07 for asphalt
A_R=50(mm)*60(mm) in our system
R=10m
P_L'=25W

So P_L'' = 3.76*10^{-6} W.
The Power incident on a photodiode is about P_L' / 25 = 150 nW
The typical sensitivity of APD is about 31A/W. So the responsivity current at 900nm is about 4.5uA.

1.3 Electronics

The system electronics includes 5 layers, shown in Figure 1-5:

1) Signal Processing: amplifies analog signals and filters noise, converts analog signals to digital signals
2) Device Driver: transmits signals from the circuit board to the computer. Defines how the interface operates.
3) Data Reading: reads signals from the interface and stores data in RAM or onto the hard disk
4) Computing: calculates speeds, lengths, and accelerations of vehicles from acquired data.
5) GUI: displays calculation results in a user-friendly way.

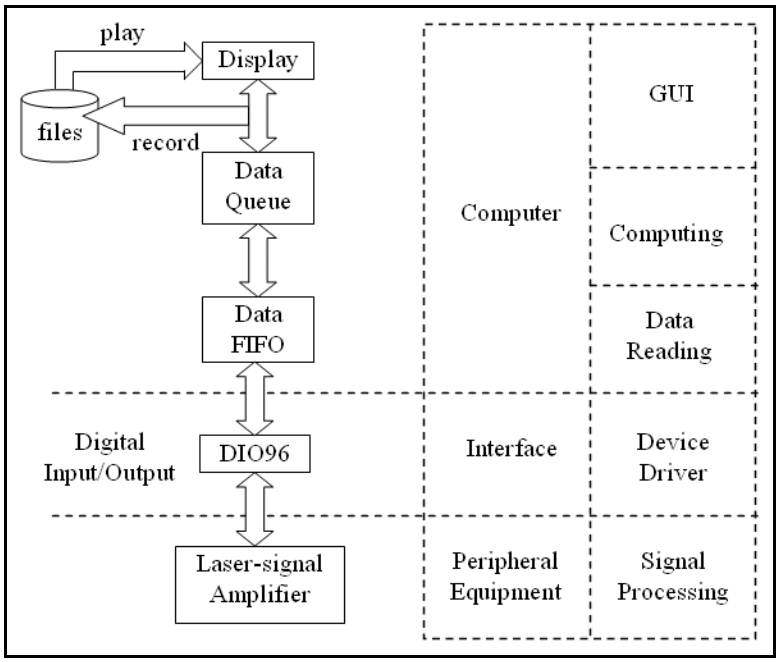


Figure 1-5: Electronic Layers of the LBDS

According to the principle of detection, the system needs only to distinguish vehicle presence under the laser lines. It is advantageous to simplify the system by providing a digital output signal from the electronic circuitry. Using a digital signal as output from the hardware will significantly simplify the signal processing in the software, and therefore reduce the computer system requirements. The implementation of this method is based on the high signal-to-noise ratio of the new circuitry. The hardware consists of five parts: the power supply, the laser components, the amplifiers, a microprocessor, and the digital I/O board with a computer.

The high voltage pulse generator for the laser diode is built into the laser unit, powered by 12 V DC. Because the pulse generator is isolated by a transformer and is well shielded, there is very low noise induced by the pulse. The power supply for the laser units is well isolated by filters as well. As a result, even though we used only one power supply for both laser and sensor electronics, there is no interference between the parts. This reduces system cost. A 25-element avalanche photodiode (APD) array is used as the sensor in our detection system. The specifications for the required APD array are located under Appendix B. The sensor converts the reflected laser light into a current signal. The sensor circuit is the main part of the electronic hardware in the detection system. In this new version of electronic hardware, some low-cost amplifier chips with suitable bandwidth, which can meet the high demand of our detection system, were chosen for signal amplification. The time response of the new circuit provides a good match to the pulse length of the laser. The high-frequency signal can be amplified effectively without oscillation. The new circuit increases the signal-to-noise ratio by a factor of 5 relative to the previous circuit. A new method using a TTL logic circuit, instead of a sample-and-hold amplifier has been used to handle the short signal pulse. Using this method, a TTL logic circuit is triggered by the amplified signal and the output is a digital signal. Usually the sample-and-hold amplifier is the bottleneck of the circuitry time response, so this method will improve the reliability of the system time response and allow us to increase the amount of signal channels (i.e.24 channels). Using a digital signal as output from the hardware will significantly simplify the signal processing in the software. The implementation of digital output is based on the high signal-to-noise ratio of the new circuitry. The new circuitry is based on simple, cheap, commonly available electronic components. Thus the cost of new circuitry is only one-fifth of that previous cost. This is vital for the commercialization of our system in the future.

The circuit can be divided into three stages: signal pre-amplification, Microprocessor-based feed-back control signal conditioning, and digital output, as shown in Figure 1-6.

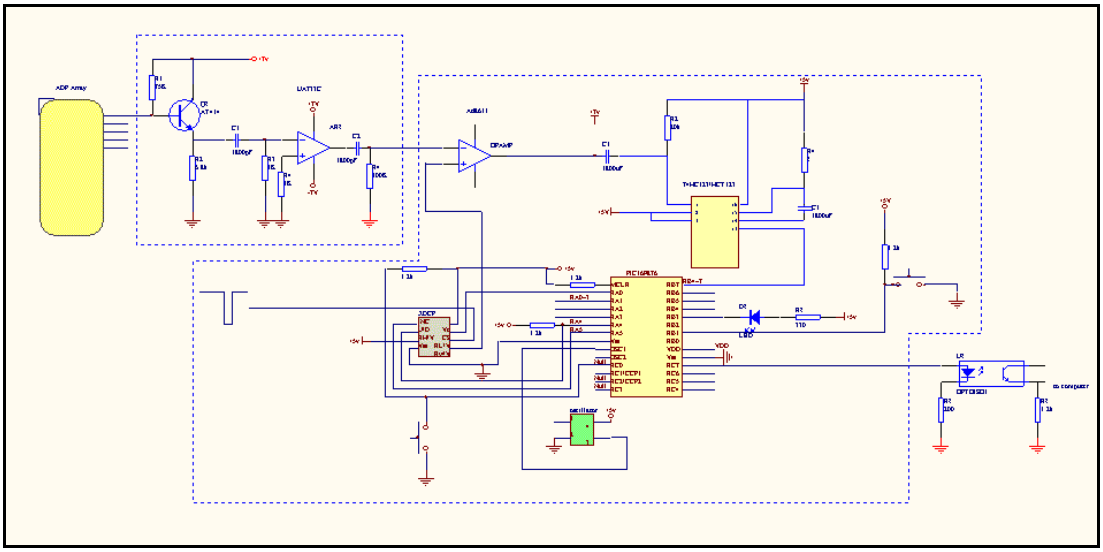


Figure 1-6: Circuitry of Sensor Electronics

Since the signal output from the APD array is a very weak current signal, a dual stage preamplifier is used to convert the current signal into a voltage signal and amplify it from a 9 nano-amp signal to a 2.5 volt signal. The voltage signal goes into a multi-vibration mono-stable oscillator circuit which outputs a 5 Volt pulse when an input pulse occurs. The output pulse then goes into a PIC microprocessor which outputs a digital signal with a high of 5 Volts to a computer when ever the laser is not blocked and a low of 0 Volts when the laser is blocked. Figure 1-7, below, shows an example of the signal conversions.

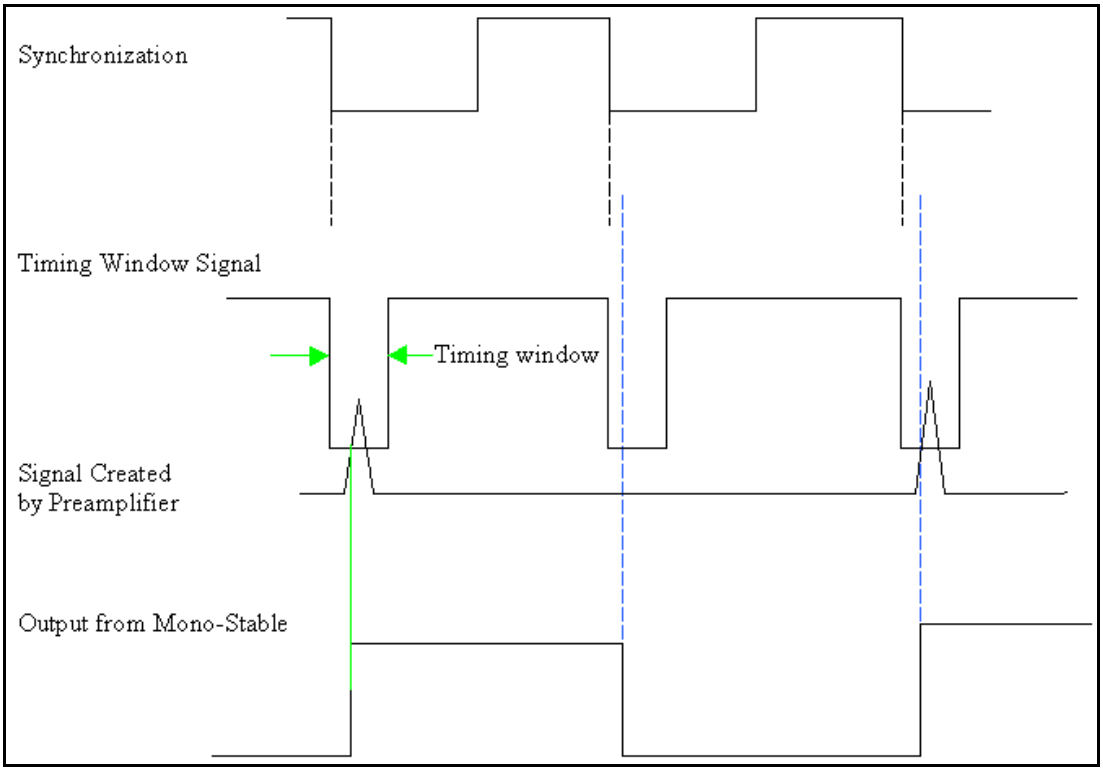


Figure 1-7: Signal Conversions

Capacitors are used to isolate DC signals between different parts of the circuit, so that a continuous signal can not pass through the circuit. Ambient lights normally generate continuous signals. After intensive testing and adjustment, the interference between different parts of the circuits are reduced to the lowest level, so that the system has a signal/noise ratio 5 times lower than the previous version. The transition time is an important factor to the accuracy and consistency of data measured.

The potentiometers (W1) are used to adjust the sensitivity of the analog circuit output. Since there are eight signal channels in the current system, manual adjustment of the potentiometers takes a long time when testing on the highway. Manually adjusting the potentiometers becomes tedious if the system is extended to 48 channels. For convenient adjustment, a microprocessor was introduced into the circuitry to automatically adjust the potentiometers. Using this microprocessor, the adjustment can be done in less than one second. We also expect the accuracy of the adjustment to be much higher than manual adjustment.

The LBDS utilizes a standard power supply that is then fed into multiple DC/DC converters. The power supplied to each side of the LBDS has been separated in order to reduce and effect each side would have on each other. The laser power supply and their timing window signals are created by a microchip. Figure 1-8, on the following page, shows the schematic diagram of the laser power supply.

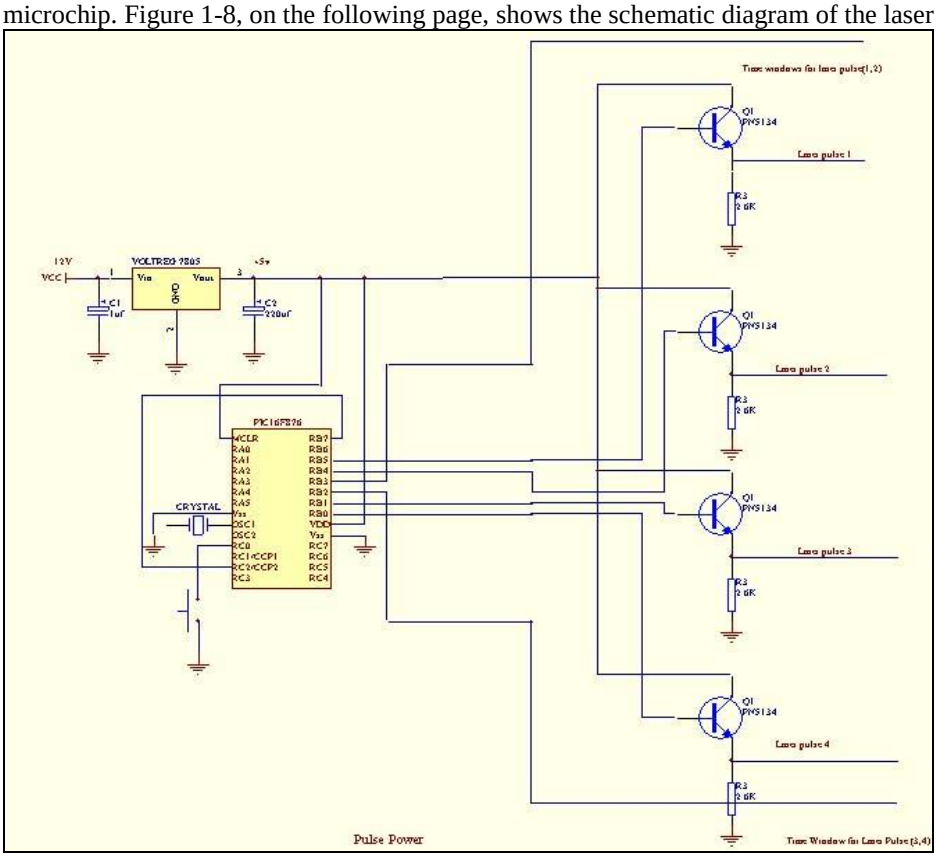


Figure 1-8 Schematic of the Laser Power Supply and Timing windows signals

The LBDS utilizes three different boards in order to take in the signal from the APD array and output the signal to the computer. All board diagrams are located in Appendix A. The main board has 6 slots and a Rabbit processor connector. The Rabbit processor implementation is for future work. The 6 slots are used for the PIC boards. The main board provides the laser source, timing window signals to keep the microchips synchronized with the pulsed laser beam. The PIC boards can easily be plugged into and pulled out of the main board. This allows for easy maintenance, repairs, debugging and expansion. One PIC board contains two PIC microprocessors, 8 digitally controlled

potentiometers, 8 reflected signal – timing window signal comparators, mono-stable circuitry, feed-back control circuits, and have end connectors for the signal output to the computer. The output from the laser system goes through the PIC boards' computer end connectors that connect to a digital input output board. Figure 1-9, on the following page, shows a circuit schematic of the PIC board.

The timing window is shown in the Figure 1-7.The signal from preamplifier is compared with a timing window signal which is controlled by the PIC microchip. The down pulse of the timing window signal will be set automatically to a proper bottom value according to the noise and the strength of the reflected signal by the PIC microchip. The AD8611 comparator will generate an input signal for the monostable circuitry. Normally, the signal out of timing-window is at 5 Volts, and the preamplifier signal is less than 1 Volt. So the output of the AD8611 is HIGH. In the timing-window, The reflected signal will come. If the signal is bigger than the bottom of timing-window, the output of AD8611 will be LOW which will cause the output of the monostable is HIGH. If the reflected signal is blocked by vehicle, the element signal from APD is smaller than the bottom of timing-window. The comparator will generate a LOW output which will cause the monostable to output a LOW.

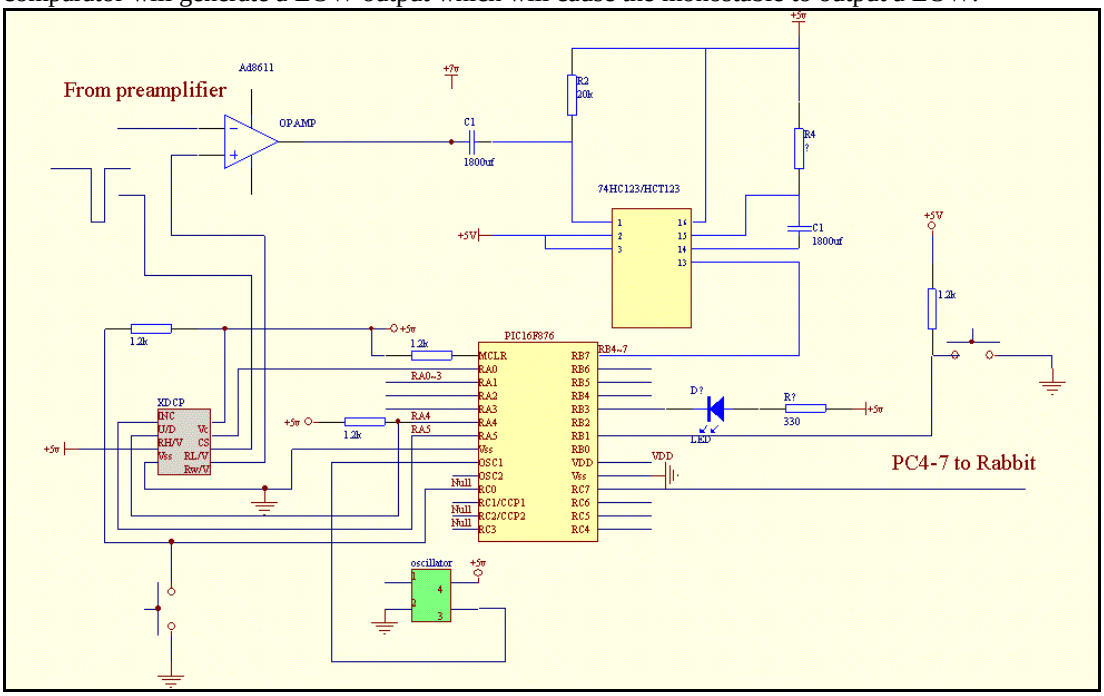


Figure 1-9: Schematic Diagram of the PIC Boards

1.4 Real-Time Device Driver and Data Acquisition Software

The Lynx OS, on which the old software is based, is a hard real-time operating system. How long each process runs for is determined by one of the scheduling policies selected (Default, Round Robin, and FIFO). For the default scheduling, a "time quantum" value is established by the constant QUANTUM, defined in param.h. The time quantum is used to determine how long each process at a particular priority level will run before giving up the CUP to the next process (at that level). As long as there are processes that are ready to run in the queue for that priority level, the scheduler will schedule them to run in a round-robin fashion, with each process running for that priority level's established quantum time.

Linux Threads provides kernel-level threads created with the new clone() system call and all scheduling is done in the kernel. Therefore Linux Threads is an implementation of soft real-time. In Linux the process priority values have a range from -20 to 20, while in the Lynx OS process priority values range from 1-255. When porting code from Lynx to Linux, the priority values should be

changed. By changing the priority values and queue sizes to fit those in Linux, all tasks can be fulfilled before the deadlines. This ensures that no data is lost.

Both the Lynx OS and Linux Threads implement POSIX 1003.1c. However the Lynx thread is based on the interfaces defined by POSIX Draft 4, while Linux Thread is based on the final standard. The POSIX Draft 4 interfaces vary significantly from the final standard POSIX 1003. The differences are described below.

In the Lynx OS, the pthread_set_prio function is used to set a thread's scheduling priority attribute. In Linux Thread the pthread_attr_setschedparam function is used to set the priority. Listing 1 is a segment of code used to initialize and setup a thread attribute object and to create a thread in Linux. Structure sched_param contains a single member that specifies the scheduling priority.

```
pthread_attr_t read_attr;               /* thread attributes */
struct sched_param read_param;         /* used by pthread_attr_setschedparam() */
.
.
.
  if (pthread_attr_init(&read_attr)<0) /* create thread attributes */
    perror("sensorGatherArray(): pthread_attr_create()"), exit(1);

  pthread_attr_setinheritsched(&read_attr, PTHREAD_EXPLICIT_SCHED);
  pthread_attr_setschedpolicy(&read_attr, SCHED_RR);
  pthread_attr_setdetachstate(&read_attr, PTHREAD_CREATE_DETACHED);

  read_param.sched_priority = getpriority(PRIO_PROCESS,0)+PRIO_READ;

  pthread_attr_setschedparam(&read_attr, &read_param);
  pthread_attr_setschedparam(&record_attr, &record_param);

  if (pthread_create(&read_tid_sensor, &read_attr, sensorThread, s)<0)
    perror("sensorGatherArray():pthread_create(read_tid)"),exit(1);
.
```

Counting semaphores are used to synchronize data read and written from queues. Lynx OS actually provides two types of counting semaphores, System V-compatible and Lynx proprietary. The later is used in our software. The following table shows the difference of the features and syntax of the counting semaphores in Lynx and Linux.

Features	Syntax	
	Lynx	Linux
Create counting semaphore	csem_create_val	sem_init
Suspend thread	csem_wait	sem_wait
Suspend thread-no blocking	csem_wait	sem_trywait
Signal semaphore	csem_signal	sem_post

Even though Linux Thread is soft real-time, the tasks which are not time-critical can still be implemented as long as the data buffer sizes are large enough. We set the buffer size twice as in Lynx and there is no observed data lost.

Some new functionality has been added into the new software in Linux system. In the old software, the real-time signals are displayed continuously, so it is hard to see the detail of signals when vehicles passing under the detection system. In the new software, a one-shot mode is used to capture the signals so that the signals will stay on the screen (until the display is reset) after a vehicle has passed the system.

1.5 Microprocessor Closed-loop Control Unit

A microprocessor was introduced into the new system to adjust the potentiometers automatically and implement signal processing. The microprocessor we chose is a PIC 16F876 from Microchip, Inc. Microchip's RISC-based MCUs are designed for applications requiring high performance and low

cost. MCUs combine high performance, low cost, and small package size, offering the best price/performance ratio in the industry. The PIC16F87X Microprocessor (MCU) family provides a migration path from OTP to FLASH in 28 to 44-pin packages, with a wide range of peripheral integration options. This family features a 14-bit instruction set, 5 to 8 channels of 10-bit analog-to-digital converters, interrupt handling capability, various serial interface capabilities, Capture/Compare/PWM, brown-out detection, and an 8-level deep stack. The PIC16F87X family provides performance and versatility to meet the most demanding requirements of analog designs. With FLASH program memory, PIC16F87X devices can also be reprogrammed over the entire operating voltage range. PIC16F876 has up to 8K*14 words of flash program memory, 378*8 bytes of data memory (RAM), and 256*8 bytes of EEPROM data memory. It can run at clock speed of 20 MHz. This Microprocessor meets the requirements of our system at a reasonable cost.

1.6 The Principle of Closed-loop Adjustment

The system uses a closed loop to control the amplifier. The schematic of the feedback controller is shown below. Here (a) is the first amplifier and β is the second amplifier. And feedback is through (b). The system output is connected to F. Assume the critical input value of trigger (0->1) is Ic. The critical value of noise, which can not trigger the D from 0 to 1, is Ln. The critical value of signal reflected from the ground, which enable the D trigger from 0 to 1, is Ls. We can get the formulation as follows.

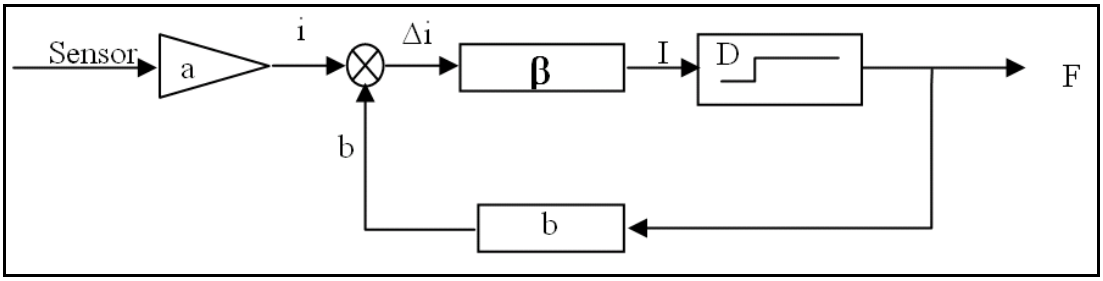


Figure 1-10: Closed Loop Diagram

$$\Delta i = i - b$$

$$I = \Delta i * \beta$$

$$F = \begin{cases} 0: \text{If the laser beam is blocked by a vehicle} & \text{-> I < Ic} \\ 1: \text{If the sensor can receive the reflected laser beam from the ground} & \text{-> I > Ic} \end{cases}$$

If F = 1, then

$$(i-b)*\beta > I_c$$

If $b < i - \frac{I_c}{\beta}$, the output F will be the high.

If $i = 0$, the output F must be: $b*\beta > -I_c$

We must set a certain limitation I_n if $I < I_n$ so that F is not triggered by noise, so the output should still be 0.

$$b > I_n - \frac{I_c}{\beta}$$

If we want to pick up the signal of laser beam reflected from the ground, for example, when $I > I_S$, the output F must be 1.

$$b < I_s - \frac{I_c}{\beta}$$

so, $b \in [I_n - I_c/\beta, I_s - I_c/\beta]$.

From the equation above, we can get the conclusion that the bigger the difference between signal of the laser and noise, the wider the range of b. It means that the system will be more reliable and steadier.

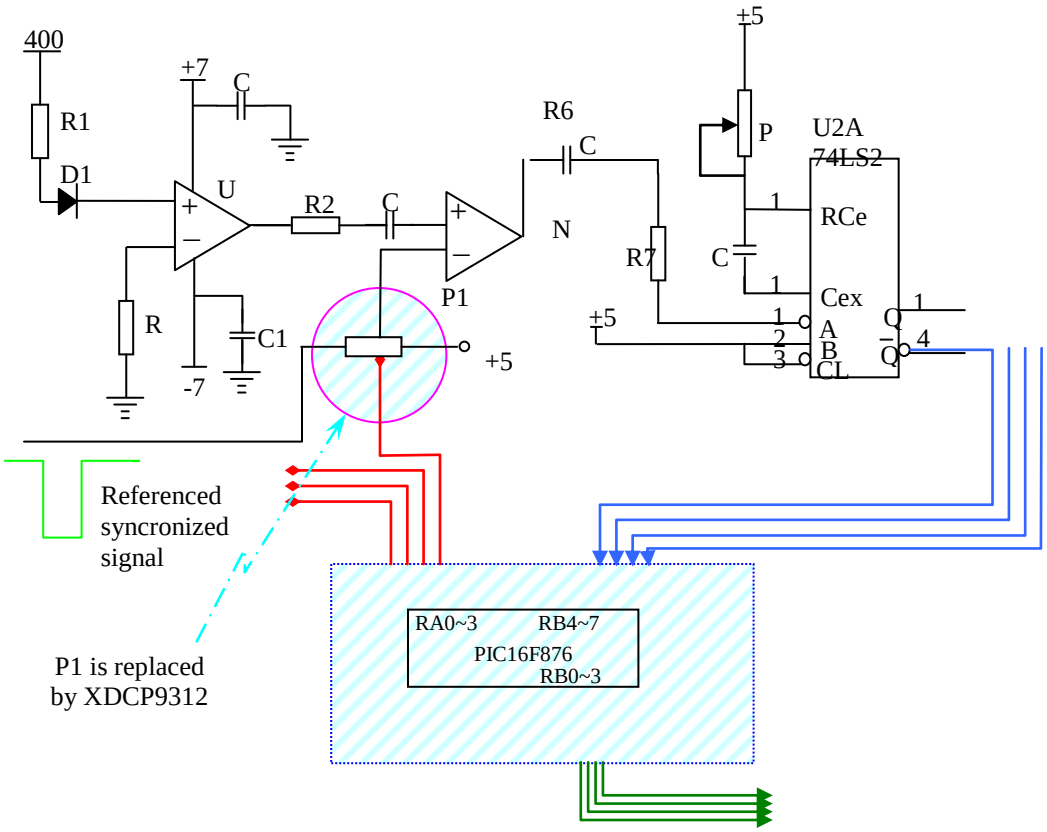


Figure 1-11: The Principle of Closed-loop Adjustment

Figure 1-11, above, shows the microprocessor controlled digital potentiometer adjustment principle integrated into the original signal amplification and processing circuitry. The potentiometer P1 is replaced by a digital potentiometer XDCP-X9312. It is a 100-step digital potentiometer controlled by

digital pulses. The digital output of 74HC123 is fed back into RB4-7 of port B in the microprocessor to adjust the potentiometer. The output signals are sampled and time-stamped. When the system is triggered to perform the automatic adjustment, the potentiometers are set to zero. The microprocessor then sends a control pulse to the digital potentiometer to increase the value of the resistor. Each microprocessor controls 4 sensor channels. The hardware and software of microprocessor adjustment unit is discussed in detail below.

1.6.1 The Closed-loop Control Unit Hardware

The detailed closed-loop control unit circuit is shown in Figure 1-12. The signals from 74HC123 are introduced into RB4-RB7 of the microprocessor. The microprocessor processes the signals and sends output pulses to the digital potentiometers to adjust their values. A switch is used to initiate automatic adjustment. A LED flashes whenever the adjustment is in progress.

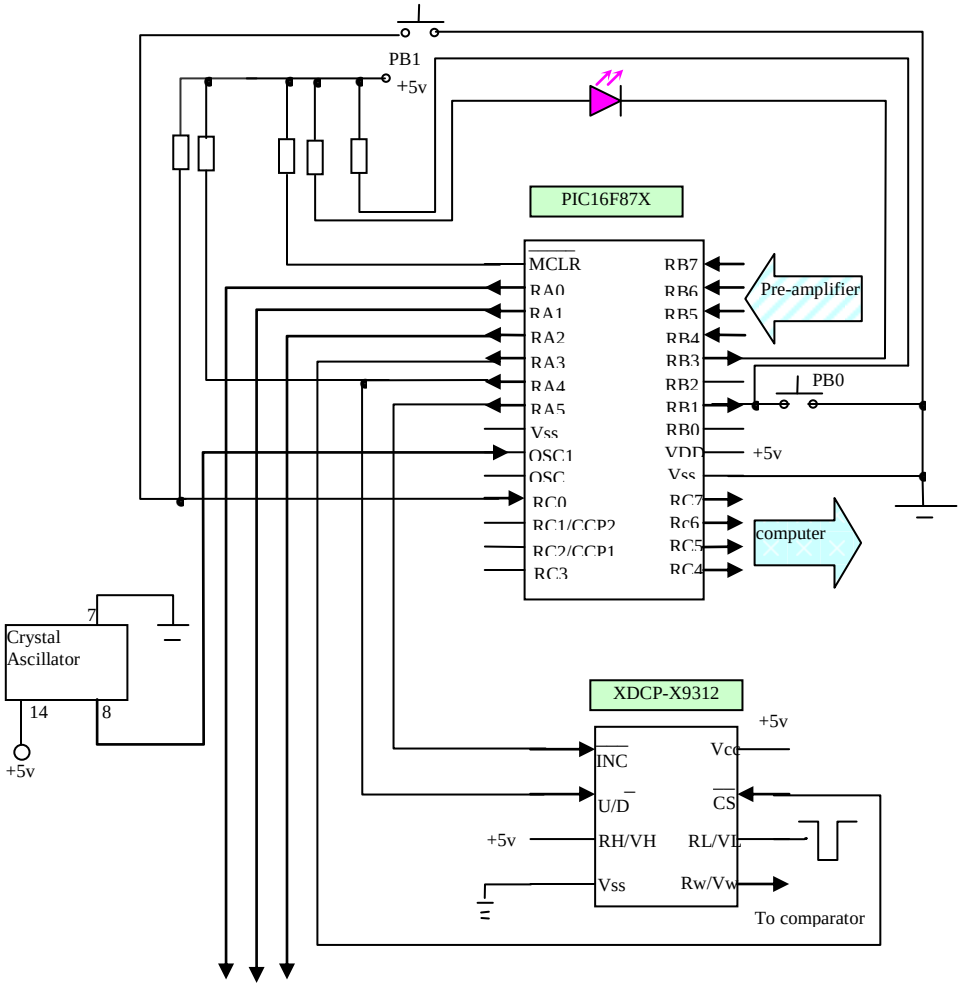


Figure 1-12: Closed-loop Control Unit Circuit XDCP

RA0~RA3 (output): cs of XDCP for channel 0~channel 3,

RA4 (output): $\overline{INC}$ of XDCP for all channels

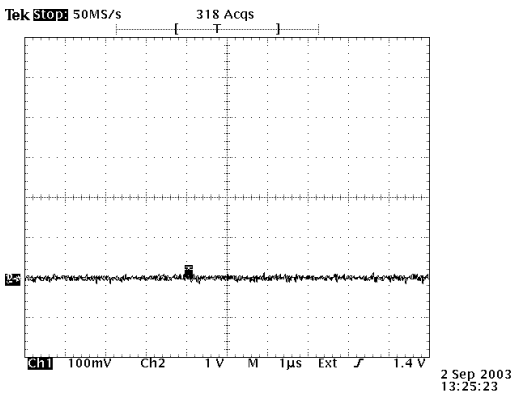
RA5 (output): U/$\overline{D}$ of XDCP for all channels. If 1, the potentiometer value is increased, otherwise it is decreased.

OSC1: CLKIN;
OSC2: CLKOUT;
RC0: If RC0 is set to 0, the system is in adjusting status; otherwise it is in work status filtering noise. It is inputted from the switcher.

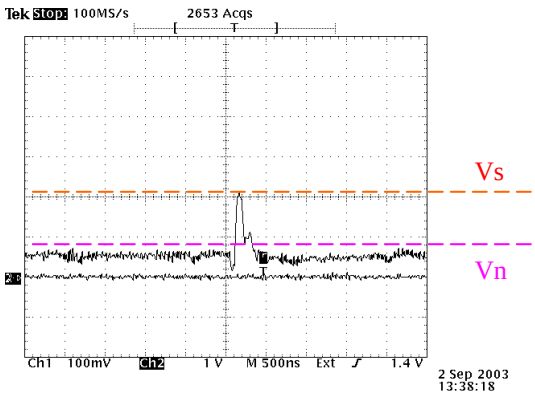
RB4~RB7 (input from sensors): feedback of channel0~channel3. A change of signal in the port causes an interrupt.
The XDCP's potentiometers resistors can be adjusted from 1000 to 100 (kΩ)

The value of the digital potentiometer is increased or decreased by one step whenever a pulse is applied to the INC pin. In the microprocessor, RA5 of port A outputs pulses to control the value of the digital potentiometers according to the input data from the output of 74HC123. RA0~RA3 are connected to CS (chip select) of the digital potentiometer to select the chip that is currently adjusted. RA4 controls the potentiometer increase and decrease when there is a pulse presented on pin INC. The microprocessor clock is provided by a 20 MHz crystal oscillator that is connected to OSC1. There are two microprocessor states, adjustment and working status. In the adjustment status, which is enabled by applying LOW to RC0, the microprocessor takes feedback from the output and adjusts the digital potentiometers to proper positions. After adjustment completion, RC0 is set to HIGH and the potentiometer values are maintained. The adjustment status is initiated by pushing a button switch, which causes an interrupt in the microprocessor. In working status, RC0 is set to HIGH and the microprocessor begins signal processing. The processing removes some unwanted signals from the vehicle reflection. The processing implementation is based on the detection of signal periods. The unwanted reflection signals are removed by determining the signal duration. If the signal duration is less than a specified amount, it is considered an unusual signal.

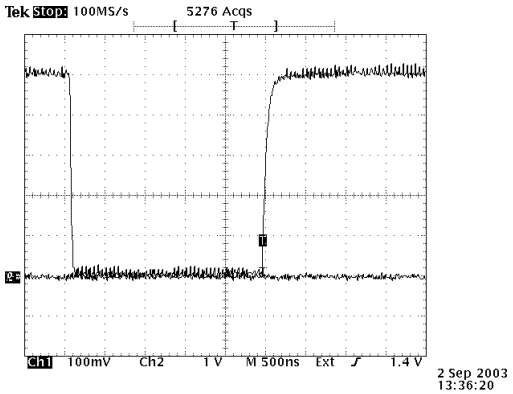
In order to filter out unexpected noise, power triggers for the laser sources have been separated. Another push button connected to PB1 is added (called BT0). When the button is pushed, the microchip will measure the amplitude of the noise and record it into memory. Another push button connected to RC0 (called BT1) is then pressed which tells the Microchip to set the referenced synchronized signal to ½ or 1/3 of Vs (signal value)-Vn (noise value) shown in the figures below.



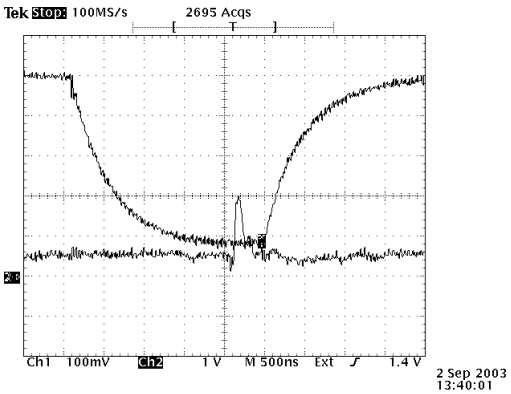
Oscilloscope View with no Signal



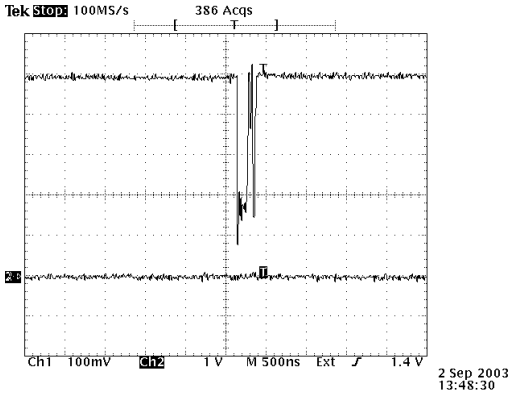
Oscilloscope View with a Signal



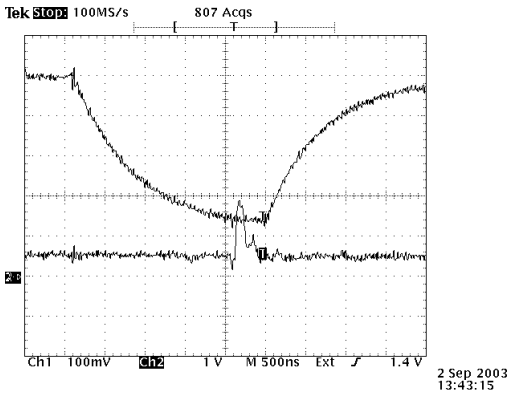
Potentiometer Signal before Adjustment



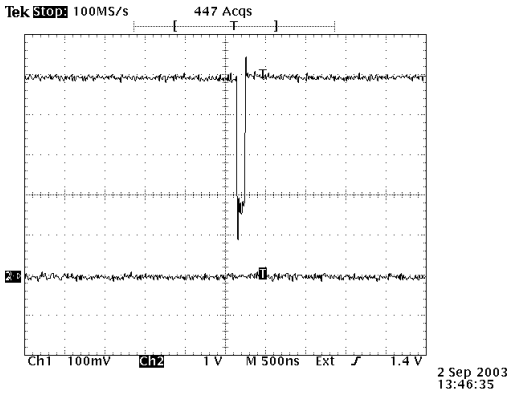
Potentiometer Signal after Pressing BT0



Output of Comparator after Pressing BT0



Referenced Sync Signal after Pressing BT1



Output of Comparator after Pressing BT1

1.6.2 Microprocessor Software

Figure 1-13 shows the flowchart for adjusting the digital potentiometers. The software is written in C. In the beginning of adjustment, all potentiometers are set to zero. When the BT0 button is pushed, the source microchip, which controls the power source for lasers and provides the referenced synchronized signals, will turn off the lasers. After a certain time, the source microchip will turn on the lasers. At the meantime, the adjustment microchip, which is used to adjust the potentiometer, will set the potentiometer to 0 and increase the resistance of the potentiometer until the feed-back signal is LOW. During this adjustment, the laser source must be off. The value of the potentiometer is then stored as BC#(0-3). This is the value of noise from the electronic circuitry board. This procedure

(called pre-adjustment) finishes in 2-3 seconds. After the pre-adjustment, the BT1 should be pushed to set the correct position of potentiometer. At first, the potentiometer is set to 0, and then the program starts adjusting by increasing the digital potentiometer one step. It then reevaluates the signal to check if the potentiometer is in the correct position, which causes the output from the monostable to be LOW. The value of potentiometer is then stored as B#(0-3). The program will reset the potentiometer as (B#-BC#)/2. When we get proper values of Digital controlled potentiometers, Microchip give a signal to lock the value of XDCP, which can be kept after power on next time. And LED the indicator will be on one of three statuses:

Normal (high): a proper value gotten

Flash (slow): value may be equal to zero (the system should be adjusted)

Flash (fast): value may be equal to maximum 10k ohm (the system should be adjusted)

Considering of keep time of potentiometer, this piece of program has some delays of time. This progression needs some milliseconds to finish.

The signal processing starts at the interrupt caused by the input status change (from sensor electronics) of the microcontroller. The processing is also based on the detection of the signal periods. The number of interrupts is stored in a register. If this number is less than a preset value FILT_NUM, the signal causing these interrupts is considered an abnormal signal and will be taken out of the output. Also, if the signal has an incorrect period, the signal is considered noise and is deleted from the output. The main modules of microprocessor program are described below. The source code is listed in Appendix D.

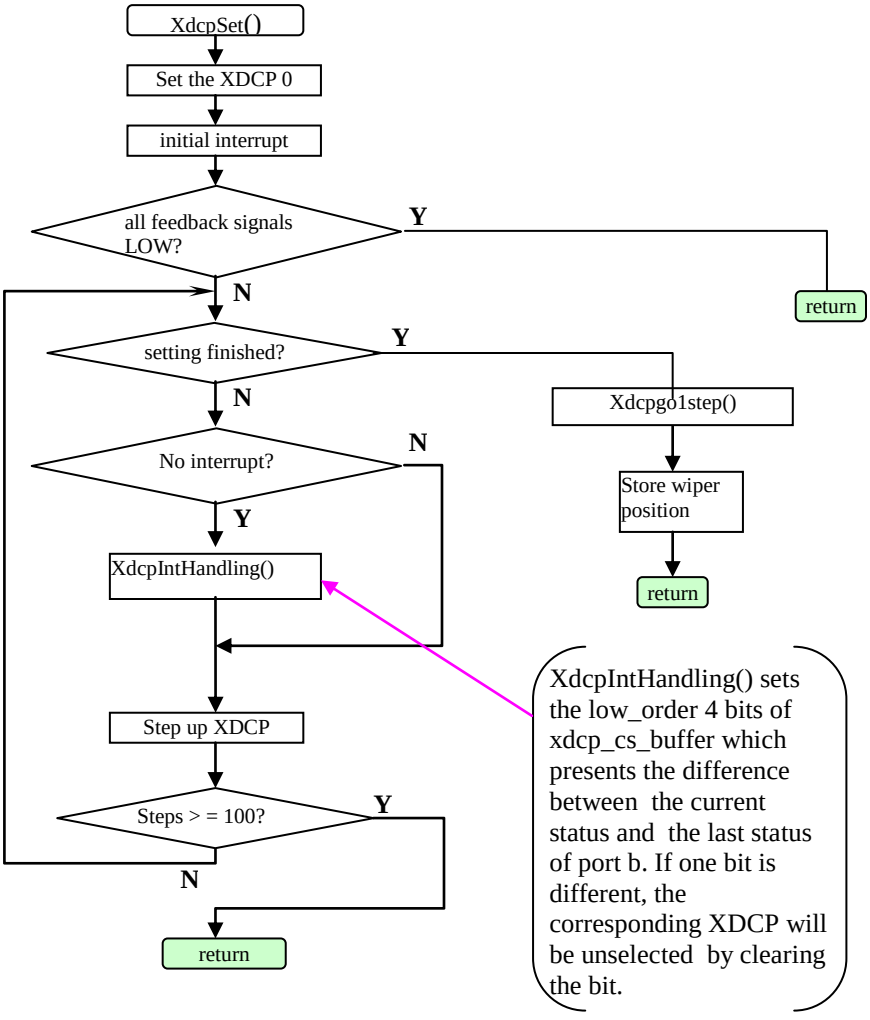


Figure 1-13: Flowchart of XDCP Adjustment

interrupt()
The interrupt handler stores the 16-bit time stamp and port B status in an array of 24 bytes, when an interrupt from port B<4:7> is received. The time stamp is stored in the first two bytes and the status is stored in the third byte. Therefore the array can store the time stamps and statuses of 8 interrupts. The interrupt procedure is better when it is shorter. It should be short enough to be processed before the second interrupt comes.

XdcpSet()
When the adjust button is pushed, this function will be called by the main( ) procedure. XDCP is set to the proper value so that laser signal can be detected by the digital output circuit. The flowchart is shown in Figure 1-13.

XdcpIntHandling()
When an interrupt is generated, the interrupt counter is incremented. This function is called by XdcpSet() when the interrupt counter is not 0. This function also reads data from the interrupt status array, determines which channel(s) cause the interrupt, and sets the channel selection variable (xdcp_cs_buffer).

FlashLed()
Outputs a pulse to the port c pin 7, causing an LED to blink.

Pwm()
Set capture/compare/PWM Register1 for pulse width modulation (PWM). Refer to PIC16F87X manual (p57) for details.

Init()
This function initializes the 16f876 microchip.

XdcpGoZero()
This function initializes the values of all potentiometers to 0. It is called by XdcpSet().

XdcpGo0()
This function is used to set the selected potentiometer to 0. It is called by XdcpSet().

XdcpGo1Step()
This function will increase or decrease one step according the U/D signal from the RA4 instead of the XdcpGoUp() in the old version.

DetectPeriod()
This function detects the pulse period. The period is obtained by averaging 8 successive periods. One period is obtained from 4 interrupts.

ReadTime()
This function is written in assembly to reduce the delay between reading TMR1L and TMRLH. Even though this delay is very short, TMR1H must be checked after TMR1H is read to ensure that there is no change in the high order bits of the register during lower byte reading.

Filter()

This function filters out noise and unusual signals. If the status of input keeps less than a certain time, the change of signal will be considered noise or as an unusual signal and the output will remain unchanged.

1.7 Mechanical Design

Figure 1-14shows the configuration of the newly designed system. The new mechanical design of the optics is flexible and allows the optical components to be adjusted to optimize alignment and focus of the laser beam. The cylindrical lenses, image lenses, and APD sensors are mounted on optical rails, which are mounted to the base plate. A holder that is adjustable in two dimensions clamps each optical component and sensor. The image lenses can be swung around an axis with an optical cell. The sensor is clamped by a ring which is mounted on a rotating lens holder. As a result the sensor can be adjusted via both translation and rotation.

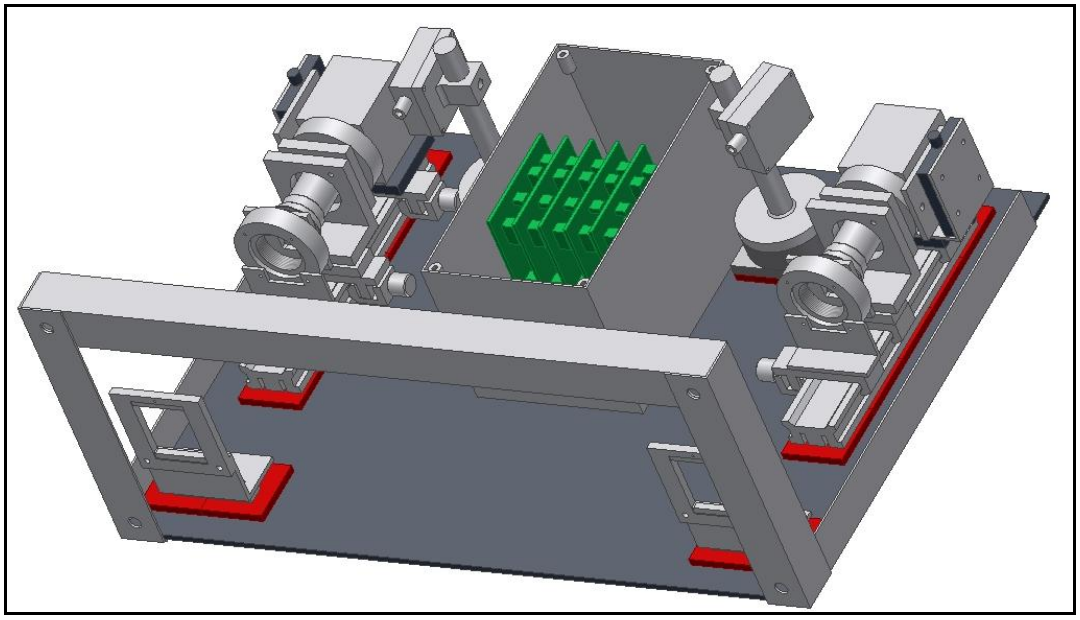


Figure 1-14: Mechanical Configuration of the Detection System

Because the new optical system is flexible, it is necessary to define a procedure to adjust the system. In the optical system shown in Figure 1-14, the adjustment includes optimizing the cylindrical telescope, and focusing and aligning optical components and lasers. A procedure for adjusting the optics of the system is as follows:

Align the telescope lenses.
Adjust the vertical alignment of the two sensor arrays.
Adjust the magnification of the optical system.
Align the laser to the optic's centerline.
Find the image focusing distance.

A clear image of the reflected laser is obtained by following the defined procedure. The optimization of the optics significantly improves (i.e. lowers) the critical height and eliminates unexpected laser reflection from reflective vehicle surfaces. A visible laser and a CCD camera are used to verify the image quality of the optical system.

The laser-based detection system we have developed will be used to identify vehicles with high accuracy. The identification data can be used to determine the real travel time of vehicles on the highway. The advantages our system has over the other systems currently used for vehicle identification, such as loop detectors [3], video image processing, and the Schwartz Electro-Optics

Autosense III sensor are easy installation, low power consumption, higher reliability, and less computation. Because our system uses active laser light, it produces its own signals to be sensed and does not suffer from limitations of lighting conditions.

In our laser-based system, vehicle lengths (bumper to bumper) are measured by the laser lines to form the vehicle outline profiles. The outline profile is used as the primary identifying feature. The system operates in the following manner**:** the basic detector unit consists of a laser and a spatially offset photo detector positioned above the plane of detection. The laser is a pulsed infrared diode laser utilizing line-generating optics, which project the laser to a flat surface where objects are to be detected. The detector consists of imaging optics and a linear photodiode array. The photodiode array receives the laser light reflected back from the region of detection. The signal from the photodiode is amplified and sent to a computer for processing. Vehicle presence is based on detection of the absence of reflected laser light. Two of these units are integrated and placed a known distance apart, allowing the speed of the object and its residence time under each detector to be measured. This data is used to calculate the object's length and compile a top-down outline profile. The details of the measurement principle are described in [2].

When a vehicle moves into a detection zone, it blocks the laser from being received by the sensor. When the first beam is blocked, the time is recorded. When the second beam is blocked, a second time is recorded. These times are used to calculate the front speed of the car. In a similar manner, when each of the beams is no longer blocked, the times are recorded and the rear speed of the vehicle is calculated. The duration that each detector is blocked is also recorded and used to calculate the vehicle length, assuming constant vehicle acceleration. In order to obtain a stable and good image on the sensors, a properly designed mechanical system is required. The mechanics should be rigid enough for field mounting and testing.

The mechanical design for the optics should be flexible and allow the optical components to be adjusted to optimize alignment and focus of the reflected laser beam. In the original system it was impossible to ensure manufacturing accuracy, because there were so many components in the system and the tolerances of some components were unknown. Even though some venders provide tolerances for some components, we were unable to determine the tolerances of ordered optical and laser components before they arrived. Even though we could get signals from this kind of optical-mechanical system, the chance of having an optimized system was very low. Thus the system did not provide a good image or necessary critical height.

There are some diffractive light and/or high-order peaks outside the width of the Gaussian profile. These diffractive side beams are hard to see, but we did observe a high order laser beam beside the main laser beam after using an IR detection card. The side laser beams are much lower than the laser peak. If the vehicle has high reflectance, significant reflection can still occur from this component of the beam. This reflection can be comparable to the reflection from the road. Accuracy is not achieved in manufacturing. All the components are adjustable in two or three-dimensions. The components are held by springs. Using these kinds of conventional opto-mechanics, stability and accuracy are easily achieved without precise manufacturing. The cylindrical lenses, imaging lenses, and APD sensors are mounted on optical rails which are mounted to the base plate. Each optical component and sensor is clamped in a mount, which is adjustable in two dimensions. The imaging lenses can be swung around an axis with the optical cell. This allows the system to be adjusted until the axis of the telescope and imaging lens are aligned.

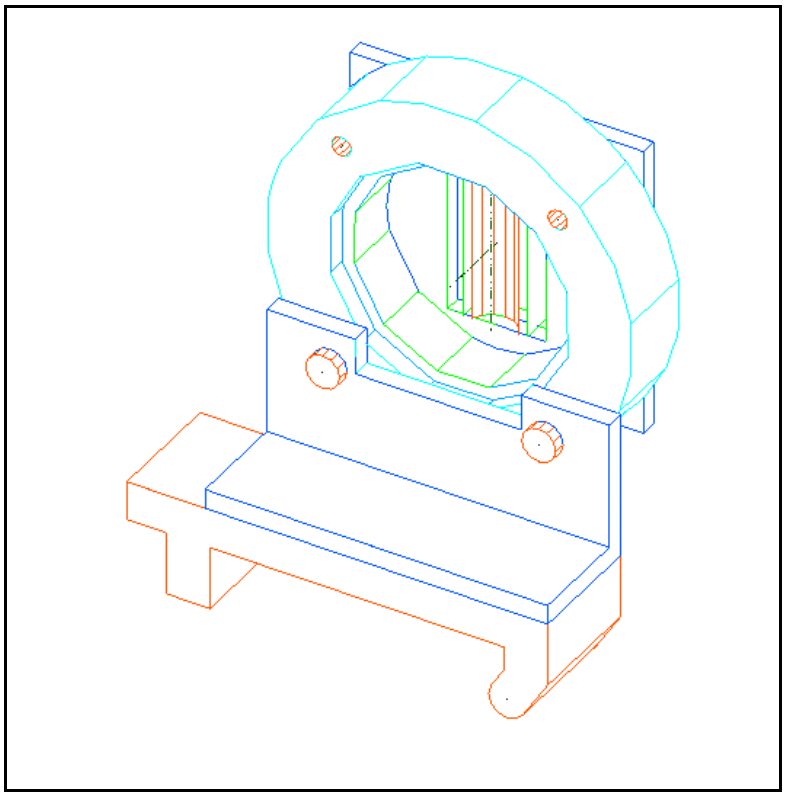


Figure 1-15: Sensor Mounting

The sensor is clamped by a ring mounted to a rotating lens holder, as shown in Figure 1-15. As a result the sensor can be adjusted through translation and rotation. This mechanism ensures the linear sensor array is perpendicular to the laser line image and is in the exact same position as the laser line. The mechanical mounting for the imaging lens is shown in Figure 1-16. The lens can swing about the X and Y-axis. The whole unit is mounted to a slide which is clamped to the rail. The lens can also be adjusted in the X direction to match the axis of the telescope.

Another area of concern is the structural rigidity of our system under prolonged mounting conditions. In favor of reducing the weight of our system, most components are aluminum. The only part that does not utilize aluminum is the structural backbone of our system. Carbon steel is used here to provide support for the plate which the system is mounted on, and to add support for the mountings. The relative positions of the mountings are found on top of this backbone, where bolts extend through both the plate and structural members. Support for the plate is crucial since any bending and twisting of the plate caused by environmental influence (i.e. wind) can greatly affect data accuracy. All system components are mounted to a metal plate.

The laser projected down to the road (9 meters) and reflected back (9m)to the system's sensor optics.. Therefore the optical path is very long. Any small change in the relative position of the laser, sensor optics, or different components in the sensor optics will cause a large shift of the image. Therefore the optical system must be adjustable after mounting. The mechanical system must be rigid enough for field use as discussed above, but the system can not be too heavy because it will be mounted over a highway. As shown in Figure 1-17, a frame was designed to support the system. It has been proven stable and the weight has not increased much.

A visible laser and a CCD camera were used to verify the image quality of the optical system. After careful adjustment of the optical system, a clear image can be seen on the screen.

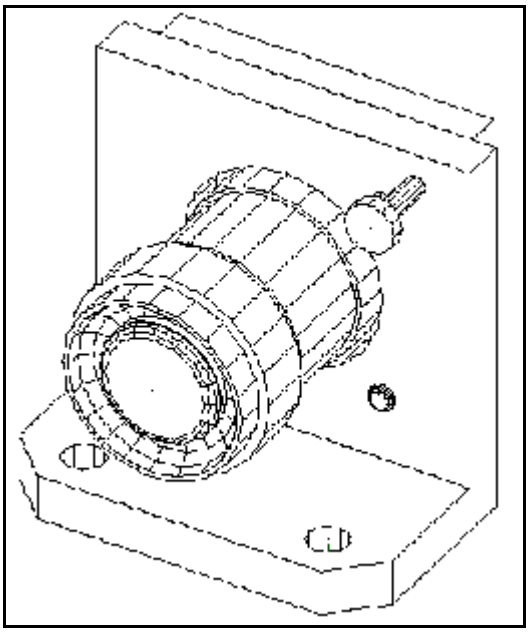


Figure 1-16: Imaging Lens Mounting

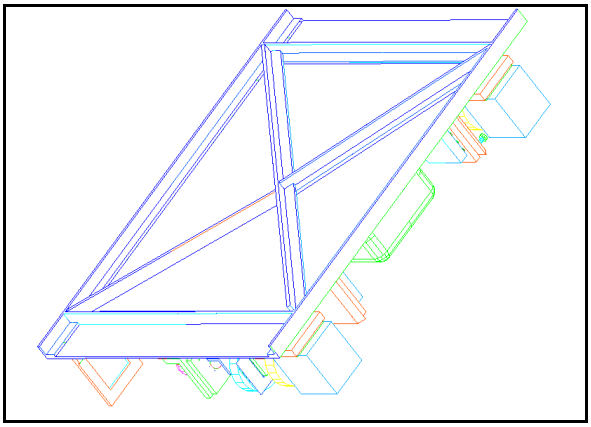


Figure 1-17: The System Support Frame and Plate

The distance between each laser/sensor pair is 60 cm. The sensors are mounted in a fixed vertical position pointing downward, and are focused on the ground forming two detection zones. The lasers are aimed towards the detection zones and mounted at an adjustable angle, allowing the system to be mounted at different heights. In this configuration the minimum detectable object height (critical height) is about 46 cm. This is lower than the bumper height of most common vehicles.

Another design element that might be taken into future consideration is reduction of the back support plate to lower wind stress. This can be done with perforations on the board, or removal of the board completely. Solely using the steel frame had been considered, but mounting problems could not be solved. One solution might be to separate the board into three portions and have empty slots between the portions. The three portions would each contain mounts for the main system components. Thus with less surface area, wind stress could be further reduced.

1.8 Operating System Platform and User Interface

Data acquisition software is used to collect, process, and display data from the detection system. Periodically data has to be read from the electronic sensor output. All tasks must be done simultaneously. However, the data has to be acquired in the same interval from the hardware. In other words, the data acquisition task should be deterministic; otherwise the information will be lost. The concurrence of the deterministic tasks requires real-time performance of the operating system.

We chose a real-time Linux platform for our new system. Pt-Linux is becoming increasingly popular in real-time embedded systems. it is one of various real-time systems and was developed by Victor Yodaiken and Michael Barabanov from the Department of Computer Science at the Institute for Mining and Technology of New Mexico. RT-Linux builds a small kernel directly over the processor, which is independent of the Linux kernel. The Linux kernel runs on top of this kernel and shares the processor with other RT-tasks. Because of its small kernel, RT-Linux has exceptional real-time performance, as long as real-time tasks are not overly complex. In our case, the most critical task is to read data from I/O ports in an accurate interval. This task is suitable for RT-Linux. In RT-Linux, device drivers are kernel modules that always run before user space processes. While data acquisition is executed in kernel space in real-time mode, the data processing and GUI program can run in the regular Linux OS. Since the outputs of the sensor circuits are digital signals now, the A/D converter is no longer needed. A general-purpose low-cost digital I/O board (PCI-DIO-96 from National Instruments) is used as an input interface between the sensor circuitry and the computer system. This digital I/O has 96 channels of configurable input/output and will be suitable for our system when all 48 channels of signal are required.

2 Problems with the Old System

2.1 No Expansion Ability

The previous LBDS had no ability to expand to more channels. This limited the potential of the LBDS to only 8 channels reducing the resolution of the detector.

2.2 Low Signal to Noise (S/N) Ratio

The feedback loop was too sensitive for the digitally controlled potentiometer (DCP). This caused the S/N ratio to be too small for highway testing. When the reflected signal is small, the DCP will be set to a small resistance in order to pick up the signal. At this time, one step is too rough to condition the small signal. The DCP has 100 steps with 100 ohms per step. If the resistor value of the DCP is set to 100 ohms, one step will cause a big error.

2.3 Same Laser Power Supply

The previous LBDS utilized one laser power supply for all of the lasers. This allowed the lasers to affect each other.

3 Testing

The system has been tested outdoors and on the highway (I-80). The procedures to line up the lasers and adjust the electronic system are as follows.

3.1 Lab Testing

The lab testing procedures are as follows:

1) Check for a laser beam and for appropriate position on the wall. If the laser line can not be seen, inspect the laser module opening first. If the laser can be seen in the opening, check if the laser line is being blocked by an obstacle. If the laser can still not be seen in the module opening, check the clock circuit and the 12V DC power supply.
2) Increase the aperture of the image lens past half way.
3) Attach the oscilloscope probe to the output of the amplifier (pin 8) and scan the laser by adjusting the screw on the rear of the laser mount until the maximum signal is reached.
4) Reduce the aperture until the width of the pulse is about 100ns and not saturated. .
5) Adjust the transistor bias to the lowest level possible, as long as the 74HC123 can be triggered continuously. The output of the digital signal can be checked in both the output of the 74HC123 or through the user interface of software.
6) Start the data acquisition software. The procedures to run the software can be found in the software section.

3.2 Outdoor Highway Testing

Field tests have been conducted near the junction of highway I-5 and I-80 in Sacramento. Figure 3-1 is a picture of the test site and the detection system mounted on the highway overpass.

Figure 3-1: Detection System Mounted Above the Highway

The results Figure 3-2 is from the highway test. A digital video camera was used to record the vehicle passing through the detection system. The picture was downloaded from the camera via an IEEE1394 fire wire port. The user interface window is shown on top of each figure. This window contains a menu bar, a status window, and a strip chart. The menu bar is used to control the system, i.e., to start or stop data acquisition and to set parameters of the system. The signals from the electronic sensors are displayed dynamically in the strip chart, which scrolls from left to right. The signal is at a high level when a vehicle is not present, and changes to a low level when a vehicle blocks the laser. This transition occurs over a time that is much shorter than the sampling interval. Since all the signals are displayed at the same position, only one signal line can be seen on the chart when there is a vehicle blocking the laser lines (or no vehicle). Signals from different sensor elements can be distinguished when they change from a low to high level (or inversely) at different

times. The front speed (v1), rear speed (v2), acceleration (a), and length (l) of the vehicle are displayed on the lower part of the window. Each line corresponds to one pair of sensor elements. Currently we only display four pairs of sensor elements.

As shown on the left side of the strip chart in Figure 8-5, the transition edges for the vehicle's front did not occur at the same time. The differences occur because of the curved front bumper on this vehicle. Different parts of the curved front bumper hit the laser (which is planar) at different times. On the other hand, the rear bumpers are essentially flat in this figure and the transition edges for the rear bumpers (in the right side of strip chart) transit up at nearly the same time. These behaviors were observed with other vehicles that had straight or curved bumpers, or when a vehicle changed lanes. The vehicle did not completely block the signal in the left pair of sensor diodes, so there were many transitions in the signal for this pair (this pair corresponds to the third row of vehicle parameters). The vehicle parameters for other sensor pairs were still accurate.

These results show that the signals for the new field prototype system are clear and the transition is fast enough for measurement. The system is consistent in measuring vehicle lengths and speeds. A vehicle traveling at a representative speed of 65 mile/hour will cross the two laser lines in approximately 17.5 ms. As a result, the error caused by the sampling interval is less than 1.14%. Misalignment of the laser lines on the road might also cause some errors, but such errors can be accounted through system adjustment and calibration.

Figure 3-2: Test Results for a Passenger Car

4 Error analysis

4.1 Error Sources of Time Records (t_1, t_2, t_1', t_2')

The errors associated with measurements of time are mainly caused by the electronics, i.e., the frequency of the pulsed laser diode and the sampling rate of the computer. We discuss these error

sources in the following sections.

4.2 The Effect of the Laser Pulse Frequency f_p

The frequency of the laser diode is 10 kHz in the LBDS system. The time error Et will be caused when the vehicle arrives at the position "enter" shown as Figure 4-1. Assume the vehicle arrives in the detection zone at the time T0. If the pulsed laser beam is dark at that time, the vehicle presence will not be detected until the time T1. As a result, the maximum error for Et is:

$$E_t = +(T_1 - T_0) = +\frac{1}{f_p}.$$

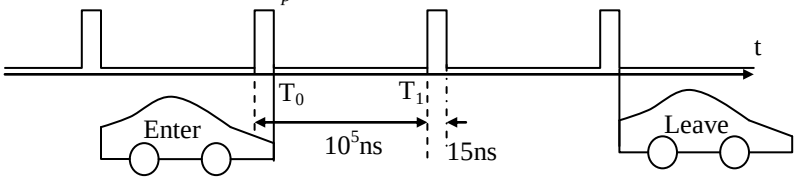


Figure 4-1: Pulse Frequency Error

Since the pulse-to-period ratio is 15/100000, the time difference T_1-T_0 can be considered to be the period of the pulse. This error is considered positive, because the actual time is larger than the detected time. The same error will be generated when the vehicle leaves the detection zone at the position "leave". But this error is negative since the actual time is smaller than detected time. So the maximum of total error caused by the frequency of the pulse is:

$$E_p = E_t \approx 1/f_p$$

Assume the speed of a vehicle on the highway is V and the length of the vehicle is L. The time τ that it takes a vehicle to pass through the detection zone is □ = L/V. We will define the error-time ratio ξ_1 = E_p/t = $V/(Lf_p)$. If we use the characteristic values V = 70.0 (MPH) * 1609 (meters/mile) /(60*60 (seconds/hour)) m/s and L ∈ [3, 25](m), then it is found that $\xi_1 \in [1.04*10^{-3}, 1.25*10^{-4}]$. These results also indicate that higher frequencies lead to smaller errors.

4.3 The Effect of the Sampling Rate f_s

The input signal of the digital input-output board (DIO96) is shown in Figure 4-2 . The computer will sample the signal at the frequency f_s. Similarly, the time error E_s is:

$$E_s = (T_1' - T_0')_t = 1/f_s$$

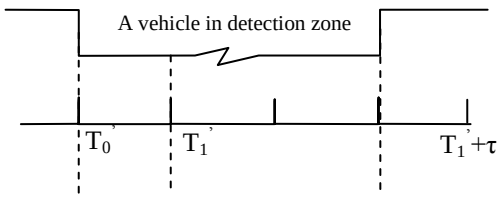


Figure 4-2: The Signal Wave vs. Sampling Wave

The frequency of vehicle signals can be estimated based on the following assumptions: the shortest

length of a vehicle is 3 meters and the following distance leads to times between vehicles that are at least 3 seconds [12]. As a result, the highest signal frequency (f_v) for vehicle signal detection on the highway is as follows.

$$f_v = \frac{1}{\tau + 3} = \frac{1}{2.2374\frac{L}{V} + 3} = \frac{V}{2.2374L + 3V}$$

where (1MPH) = (1 miles per hour) = (1069/60*60) (m/s) = $(2.2374)^{-1}$ (m/s)

The maximum value for f_v is 0.333. According to Shannon's Sampling Theorem, the sampling frequency f_s (about 10 kHz) used in LBDS system is fast enough ($f_s \gg 2\ f_v$). Because $f_s = f_p$, the error is the same as the error caused by pulse. The error-time ratio caused by the sampling rate is $\xi_2 = \xi_1$

4.4 Error Sources Related to the Distance between Two Laser Lines

The reference distance D_{ref} between two laser lines is another important parameter for calculating speed and length of vehicle. The value of D_{ref} is used in the software. However, this value may be changed by adjustment of the opto-mechanical subsystem and the distance between the LBDS system and the detected objects. Relevant error sources are discussed below.

4.5 The Effect of a Deviation Angle α of the Laser Beam

Ideally, the reflected laser light path should be perpendicular to the roadway as shown in the dashed line in Figure 4-3. In reality, however, machining error or assembly error will cause a deviation angle α.

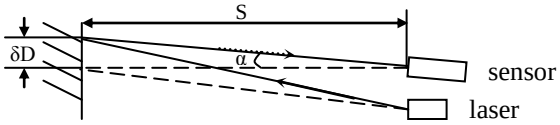


Figure 4-3: Deviation Angle

The D_{ref} indicated in the figure above will be changed to $D_{ref} \pm 2\delta D$ where $\delta D = S \tan \alpha$.

The error-distance ratio ζ_1 is:

$$\zeta_1 = \frac{2\delta D}{D_{ref} \pm 2\delta D} = \frac{2S \tan \alpha}{D_{ref} \pm 2S \tan \alpha}$$

In the LBDS, the angle α is related to the tolerance of machining and assembling.

4.6 The Effect of the Unmatched Sensor Detection Points

The detection points of one pair of sensors should be ideally at the position shown as the dashed line in Figure.7 A (this line is perpendicular to laser line). But in reality, it is possible that the detection points of one pair of sensor are located at the position illustrated by dotted-dashed line in Figure 4-4 A.

In this case, if the front or rear of the vehicle is straight, the distance (D) between two laser beams is still D_{ref}. If the front or rear border of the vehicle is curved, D will be changed to $D_{ref} \pm \Delta D$. We can develop a relationship for ΔD as shown below.

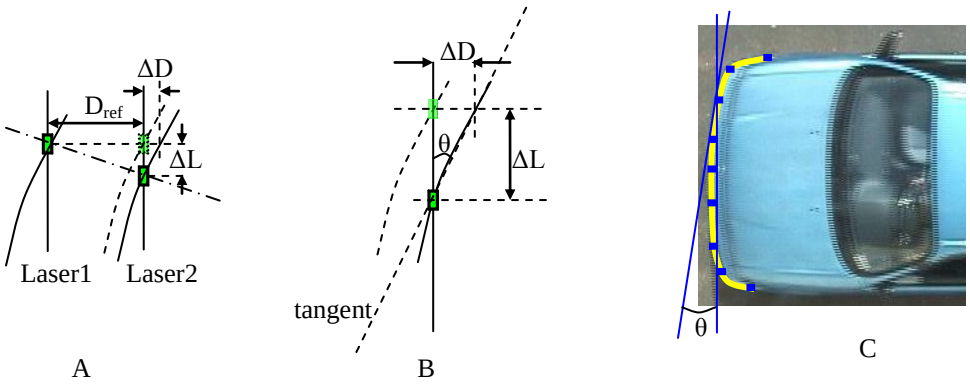


Figure 4-4: Sensor positions

$$\Delta D = \Delta L * \tan\theta \ .$$

Here, θ is the angle between the tangent line and the laser line shown in Figure 4-4 B. Figure 4-4 C is the front of a vehicle.

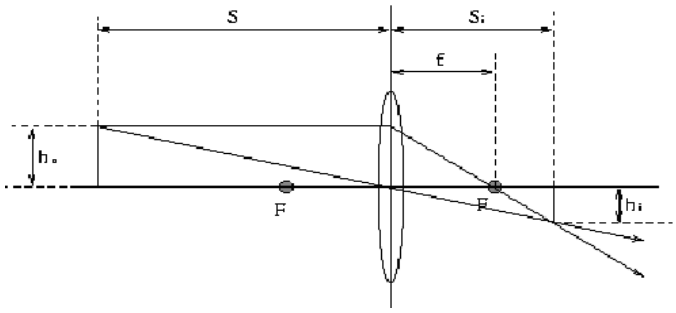


Figure 4-5: Imaging of Laser

The purpose of the optical sensor is to image the large laser line onto the smaller active area of the photodiode array. Suppose ΔL is the detection point deviation of one pair of sensors. The corresponding image Δl will be very small. The focal length of the telescope is f and the distance to the object is S as shown in Figure.8. According to the imaging principle, we can obtain [13]:

$$\frac{\Delta l}{\Delta L} = \frac{f}{S - f},$$

$$\Delta D = \frac{S - f}{f} \bullet \Delta l \bullet \tan\theta \ .$$

Generally, S is about 7~9 m, and f is only 16 mm. As a result, very small Δl values will cause large ΔL values.

For f << S , ΔD will be:

$$\Delta D = \frac{S\Delta l \tan\theta}{f}.$$

So the error-distance ration ζ_2 is:

$$\zeta_2 = \frac{\Delta D}{D_{ref} \pm \Delta D} = \frac{S\Delta l \tan\theta}{fD_{ref} \pm S\Delta l \tan\theta}.$$

4.7 Total Error Analysis

Considering all of the above factors, the outputs of the system (speed and length) are illustrated below.

4.7.1 The Speed Error

The actual speed (V_{real}) of the vehicle is

$$V_{real} = \frac{D_{ref} \pm 2S\tan\alpha \pm \dfrac{S\Delta l \tan\theta}{f}}{T_{ref} \pm \dfrac{1}{f_p} \pm \dfrac{1}{f_s}}$$

$$= \frac{D_{ref} \pm 2S\tan\alpha \pm S\Delta l \tan\theta / f}{D_{ref} \pm \dfrac{2V_{ref}}{2.2374 f_p}} V_{ref}\,(MPH) \quad (f_p = f_s)$$

Where $V_{ref} = D_{ref}/T_{ref}$ (here T_{ref} means the detected time: $(t_2\text{-}t_1)$).

The effects of varied factors to the speed are shown in Figure 4-6 as below.

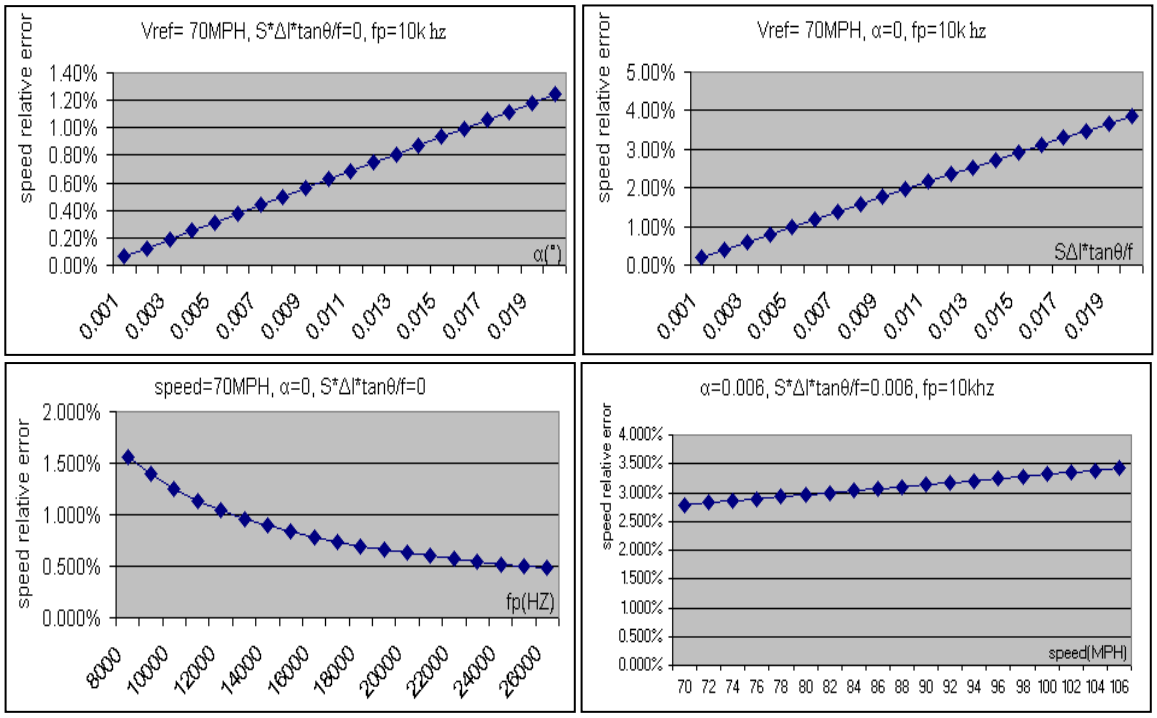


Figure 4-6: Speed Effects

4.7.2 The Length Error

Assuming the detected length of vehicle is L_D, the real length of the vehicle is L_R. According to the result for the D_{ref} and the $T_{ref,}$ we can obtain as follows (V_{ref1} is the front V_{ref}, V_{ref2} is the rear V_{ref}):

$$L_R = \overline{V}_{real} \times \overline{T}_{real},$$

$$\overline{T}_{real} = \overline{T}_{ref} \pm 2/f_p = L_{ref}/((V_{ref1} + V_{ref2})/2) \pm 2/f_p,$$

$$L_R = \left(\frac{(D_{ref} \pm \varepsilon_1 \pm \varepsilon_2)V_{ref1}}{D_{ref} \pm 2V_{ref1}/2.2374 f_p} + \frac{(D_{ref} \pm \varepsilon_1 \pm \varepsilon_2)V_{ref2}}{D_{ref} \pm 2V_{ref2}/2.2374 f_p} \right) \left(\frac{L_{ref}}{V_{ref1} + V_{ref2}} \pm \frac{1}{f_p} \right),$$

where:

$\varepsilon_1 = 2S \tan\alpha$

$\varepsilon_2 = S\Delta l \tan\theta / f$ °

The length errors caused by α, SΔl tanθ/f, f_p, speed and L_{ref} are shown in Figure 4-7.

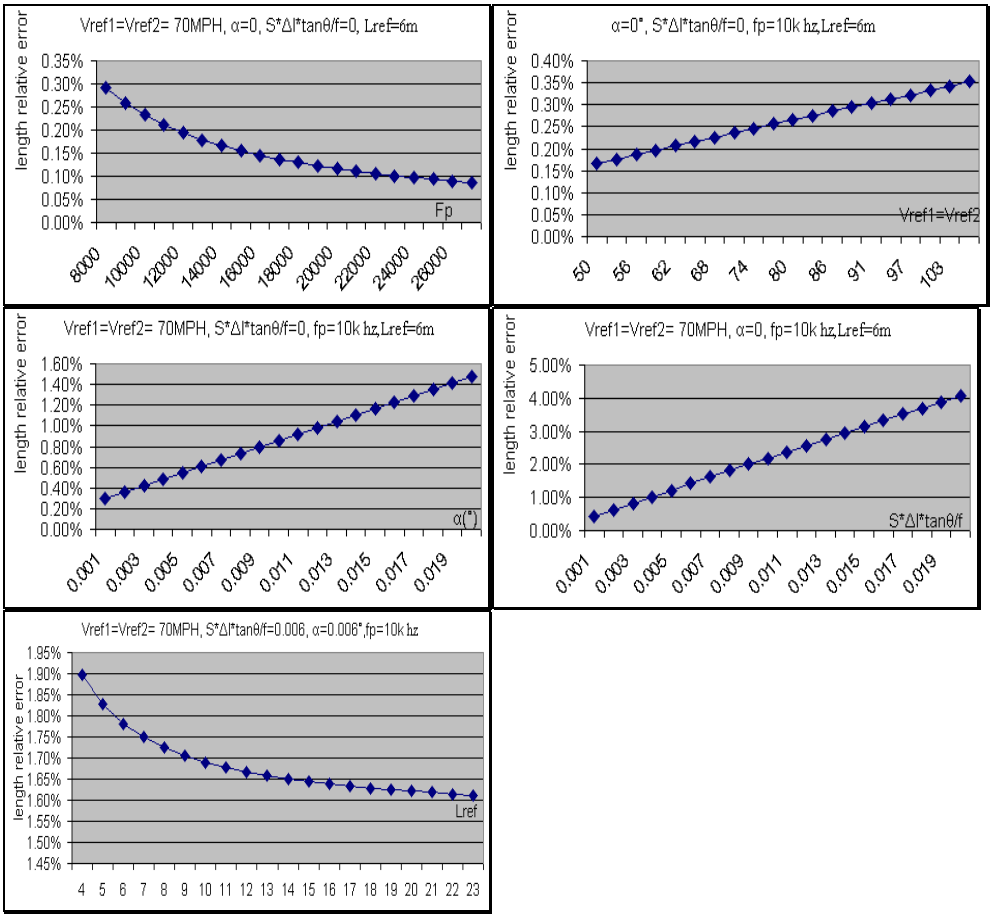


Figure 4-7: Effect on Length

5 Optical Detection of Trace Gases on the Roadway

Detection of particulates and trace gases along roadways is of interest for a number of reasons. For example, vehicles can emit one or more pollutants as a result of their operation. A motor vehicle burning a hydrocarbon fuel will generally emit species such as CO, NO and NO_2 as well as unburned hydrocarbons and other species. The species CO, NO and NO_2 are important pollutants that need to be minimized in vehicle emissions. Different vehicles will emit different amounts of these species depending on their age and physical condition, and it has been suggested that most pollutants from vehicles actually come from a small fraction (about 10%) of the vehicles on the roadway (so-called "gross polluters"). As a result, it is useful to be able to monitor emissions from vehicles under roadway conditions, which may allow identification of the gross polluters.

This report discusses potential optical techniques that may potentially be used for detection of trace gases such as CO, CO_2, NO and NO_2 on the roadway. These gas species are relevant to detection of highly polluting vehicles. The focus of this report is on optical techniques that might be used in an overhead detector situation, with signal reflection from the roadway.

5.1 Absorption Spectroscopy

Detection of specific gas-phase chemical species via optical methods is generally accomplished by subjecting a region of interest to light and then measuring the characteristics of the light that is scattered or transmitted. Some methods for detection of species include absorption techniques, fluorescence techniques and Raman measurements. A literature review (not presented here) of these techniques has suggested that absorption measurements may be most readily deployed in an overhead detector configuration. As a result, we will focus on absorption spectroscopy.

Utilizing absorption spectroscopy for detection of specific chemical species involves measuring the attenuation of a light beam as it passes through a test section containing the species. For example, it is known that molecules absorb electromagnetic radiation in well-defined bands. This behavior has been exploited in many instances to provide a method of identifying both the identities of chemical species as well as their concentrations. The identity of the chemical species can be determined by measuring one or more peak absorption wavelengths, λ_{max} (i.e., λ_{max} is the wavelength where absorption is maximized in an absorption band) while concentrations can be determined by measuring the attenuation of the beam. Attenuation measurements are typically made at the λ_{max} wavelengths, and Beer's law is employed for calculation of concentrations. Because concentrations may vary along the light path, this type of measurement typically provides average concentration measurements along the beam path.

Absorption spectroscopy methods are generally useful only when the absorption bands of the species of interest are known. A major task in applying absorption spectroscopy methods for detection of a particular species is to select an absorption band that satisfies the following criteria: (1) no interference from other species; (2) a laser is available at the appropriate wavelength; (3) the optics is not prohibitively expensive; and (4) a detector (to collect the scattered light) is available. For many molecules of interest, absorption bands have been extensively investigated and may be considered fairly well known. As such, it is usually possible to select specific bands that should be free of interference from other species. For example, it is known that CO and H_2O, which are both present in vehicle exhaust, can have some overlap in their absorption bands. However, is has been shown that CO can be detected without interference from H_2O absorption bands by selection of the appropriate laser wavelength.

Table 1 lists typical absorption wavelengths and detection sensitivities that have been used for sensing of various gas molecules. These data assume the optical path length (i.e., the path length over which absorption occurs) is one meter, which should be applicable to vehicle emission situations. In addition, these data assume that the minimum detectable laser attenuation is $1 - e^{-5} = 0.9933$, which has been found to typically apply in practice. Two wavelengths are given in Table 1 for each molecule. The shorter wavelength is in the near IR, which is easily accessible with conventional diode lasers while the longer wavelength is in the mid-IR, which can be accessed with more recent technology such as quantum cascade lasers. It is noted that researchers in this field have also used other absorption wavelengths for gas sensing. The data in Table 1 show that significantly better detection limits are generally achieved in the mid-IR (it is also noted that better differentiation between species is also generally achieved in the mid-IR).

5.2 Solar Radiation Interference

If trace species are to be measured along a roadway by using optical methods, it is important to consider the potential effects of solar radiation. The solar spectrum contains small amounts of power for wavelengths longer than about 2600 nm; this is advantageous when detecting small optical signals. For example, roughly about 3% (or less) of the total solar power exists at wavelengths longer than about 2600 nm, where this calculation is based upon blackbody radiation functions. It is also noted that the solar radiation heat flux at the edge of the earth's atmosphere is about 1353 W/m^2. Using this as a baseline value, we can estimate that the solar radiant energy flux in the representative wavelength range 2300 - 2302 nm (which is wider than what is actually needed for an absorption measurement) is about 0.012 mW/cm^2. For the wavelength range 4600 - 4602 nm the corresponding radiant energy flux is 0.0011 mW/cm^2.

For a 10 mW laser spread out over a rectangular area 1 cm wide and 3 m long, the corresponding laser power flux is 0.033 mW/cm^2. As a result, solar noise should be manageable in these measurements, especially if mid-IR lasers are employed. It is noted, however, that even if solar noise amplitudes were significantly larger than the signal amplitude, lock-in amplification techniques could be used to detect very weak modulated signals. It is also necessary to utilize interference filters with very narrow wavelength ranges so that noise from sunlight is minimized - these calculations were performed with the assumption that the interference filter transmits radiant energy only over a 2 nm wide spectral window. In addition, these calculations assumed that there is no attenuation of incoming solar radiation by the earth's atmosphere, thus providing worst-case estimates.

5.3 Laser Characteristics

It is also important to consider some characteristics of lasers. For a laser to be useful in a highway situation, it should be compact, reliable, controllable and should not require very much power to operate. The laser system should also be eye-safe. Gas and dye lasers generally do not meet all of these conditions, while solid-state lasers do meet these criteria. As a result, we will concentrate out attention on solid-state lasers. Table 2 shows data for some solid-state lasers that are applicable to or have been used for chemical sensing in the gas phase.

The data in Table 2 show that there are several different solid-state laser types that can be employed for gas sensing. Based on the data in this table, it is generally easiest to sense chemical species at wavelengths below about 2300 nm if room-temperature lasers are to be employed. It is noted, however, that quantum cascade lasers have the capability to access long wavelength regions of the spectrum, potentially providing greater measurement capabilities. Quantum cascade lasers can operate in pulsed mode (with pulse durations in the 10's of ns) at room temperature, though thermoelectric cooling may sometimes be needed.

5.4 Estimates of Reflected Optical Power

In order for a gas species detection system to work effectively, the reflected beam must have sufficient power to enable its detection. Estimates of the power of a reflected beam can be made by assuming that a laser beam is reflected diffusely from the roadway surface. If the incident laser power has the value P_I and the reflected laser power that reaches the detector is P_R, a relationship between P_R and P_I can be written as shown in Eq. (1).

$$\frac{P_D}{P_I} = \rho\phi\frac{A_D}{4\pi R^2} \tag{1}$$

Here, ρ is the reflectivity of the road surface at the laser wavelength, A_D is the area of the collection optics, R the distance of the collection optics above the road, and ϕ accounts for optical losses (e.g., after the reflected light enters the detector system). Using the representative values $\rho = 0.01$ (likely a minimum value), $\phi = 0.5$, $A_D = 10$ cm^2, and R = 7 m yields $P_D/P_I \approx 10^{-8}$. Therefore, a 10^{-3} W incident laser power (the lowest value in Table 2) would yield $P_D \approx 10^{-11}$ W = 10 pW as being incident on a detector and a 50×10^{-3} W laser (the highest value in Table 2) would yield a $P_D \approx 500$ pW signal. For comparison, a 25 W laser would yield $P_D \approx 250{,}000$ pW (250 nW). The current vehicle detection system uses a laser with about 25 W peak power for measurement of vehicle lengths. However, it is not clear whether this type of laser could be used for spectroscopic measurements because of the relative wide spectral width of the laser output (likely at least several nm).

5.5 Detector Characteristics

Some solid-state sensors that have been used for gas sensing applications include photodiodes (avalanche and non-avalanche), InAs detectors and IR photoconductors. In some cases thermoelectric cooling is employed to reduce noise. Table 3 shows some detector types and their wavelength ranges. All of these detectors can detect very low light levels (down to 5 pW or even counting single photons). Based on the data in Table 3, the lowest reflected power level (10 pW) should be detectable by solid-state detectors, so it appears that a reflected laser signal could be detected under roadway conditions. However, it is to be noted that this estimate assumes that all of the reflected laser power is incident upon a single detector. If the reflected power is spread across a linear array, the radiant power incident upon individual elements of the array will decrease, reducing the effectiveness of the detector. An acceptable compromise may be to spread the laser into a sheet projected downward onto the road surface. The reflected laser light can be directed onto a single detector element (e.g., an APD at shorter wavelengths or an IR photoconductor at longer wavelengths), thus providing an integral average measurement of the amount of the species being probed along the laser sheet path (as well as the path of the reflected light). For reliable detection when signal levels are weak, it should be advantageous to employ laser modulation coupled with a lock-in amplifier in the receiving electronics. This type of methodology should be applicable even when short-duration pulsed lasers are employed, provided that an appropriate low-pass filter is used.

5.6 An Overhead Trace Gas Detection Scheme

A potential method for overhead detection of gaseous chemical species on the roadway is shown in Figure 5-1. This system would employ a single solid-state laser situated above the roadway. The laser would direct the beam downward onto the road surface. The beam, which could be caused to be planar using cylindrical optics, will pass through the gases emitted by a vehicle before being reflected back to the receiving optics. A beam splitter can be employed to enable measurements to be taken

along a line of sight (i.e., within a single measurement plane). The beam splitter can also be used to provide a reference beam from the laser, allowing variations in laser output power (and possible wavelength) to be monitored.

The presence of a chemical species could be sensed using absorption spectroscopy techniques, e.g., by employing wavelength modulation for a cw laser or by controlling the wavelength of a pulsed laser. It is noted that the spectral width of the laser needs to be smaller than the width of the absorption line being scanned (this may be difficult with the laser currently being employed for measurements of vehicle lengths, as this laser probably has a spectral width of several nm). Optics would be employed to focus the laser line onto a detector, allowing for measurements of pollutant concentrations within the plane of the laser beam. The detector could be comprised of a single element, providing average measurements for the measurement plane. In contrast, an array could be employed (e.g., an APD array), allowing for spatial measurements along one direction of the measurement plane. However, use of a detector array may be difficult because of the small signal levels that are anticipated. The overhead geometry will provide accurate measurements regardless of the height or location of the exhaust pipe. In addition, this geometry will allow for measurements from the roadway to heights several meters above the road surface. Currently available systems are incapable of these types of measurements.

The specific system to be deployed depends upon the molecules to be sensed and the wavelength range that is to be investigated. Because of the higher sensitivities that are attainable at longer wavelengths (see Table 1), it may be desirable to employ quantum cascade lasers operating in the mid-IR. However, these lasers may be more expensive and the optics (lenses and windows) for this wavelength range may also be more expensive than what is required for near-IR applications. Data for transmission ranges of some optical materials are shown in Table 4. For example, fused silica (a common lens and window material) has good optical transmission characteristics over the wavelength range 185 to 2500 nm, becoming essentially opaque for wavelengths above 2500 nm. In contrast, germanium has good transmission characteristics over the wavelength range 2000 to 12000 nm.

Molecule	λ (nm) Detection Limit (ppb)	λ (nm) Detection Limit (ppb)
H_2O	1390 60	5940 2.0
CO	2330 500	4600 0.75
CO_2	1960 3000	4230 0.13
NO	2650 1000	5250 5.8
NO_2	680 340	6140 3.0

Table 1: Gas absorption wavelengths and detection limits for a 1-m path length

Laser Type	Wavelength (nm)	Operating Temperature	Maximum Power
InGaAsP	1200-2000	Room Temperature	10 mW (cw)
AlGaAs	750-1000	Room Temperature	10 mW (cw)
AlGaInP	630-690	Room Temperature	10 mW (cw)

Vertical Cavity	650-1550	Room Temperature	7 mW (cw)
InGaSb(As)	2300-2700	Room Temperature	20 mW (cw)
Lead-Salt	3000-30000	Cryogenic	1 mW (cw)
Antimonide	2000-4000	Thermoelectric (short λ) Room Temperature (long λ)	> 1 mW (cw)
Quantum Cascade	4500-17000	Cryogenic (cw) Room Temperature (pulsed)	 50 mW (pulsed)

Table 2: Some solid-state laser types, operating wavelengths, operating temperatures and powers. These lasers are applicable to or have been used for chemical sensing in the gas phase

Detector	Wavelength Range (nm)	Minimum Detectable Power
Silicon APD	300 - 1100	Single Photon
Germanium APD	800 - 1600	Single Photon
InGaAs/InP APD	1100 - 1600	Single Photon
InGaAs APD	900 - 1700	Single Photon
InAs	500 - 3800	5 pW
IR Photoconductors	2000 - 6000	5 pW

Table 3: Some solid-state optical detectors, their wavelength ranges and minimum detectable optical powers. Minimum detectable optical powers for InAs and IR photoconductors were estimated from noise-equivalent-power data provided by manufacturers.

Optical Material	Transmission Range (nm)
BK7 (Borosilicate Glass)	330 - 2100
Calcite	210 - 2300
Crystal Quartz	200 - 2300
Fused Silica	185 - 2500
a-BBO Crystal	190 - 3500
Sapphire	180 - 4500
YVO_4 Crystal	400 - 5000
CaF_2	170 - 7800
Germanium	2000 - 12000

Table 4: Some optical materials and their transmission ranges.

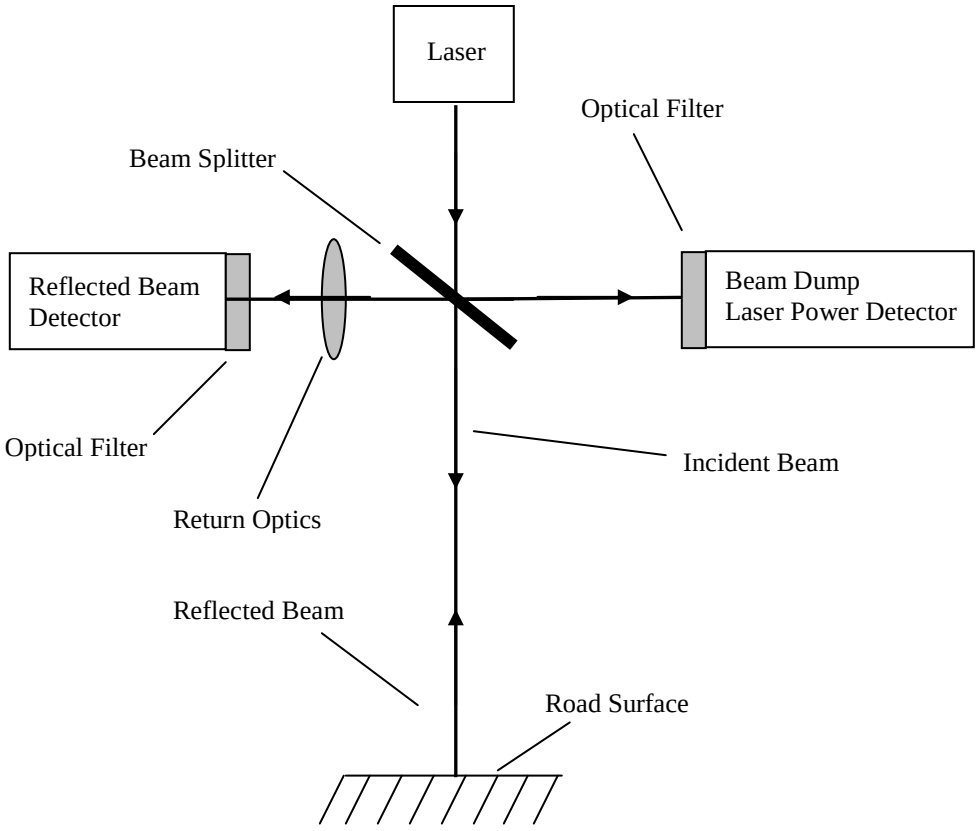


Figure 5-1: Overhead detector system schematic

6 Infrared Reflectance of Vehicle Paints

There is considerable interest in developing non-intrusive vehicle detection systems to measure unique vehicle characteristics that can be used to differentiate between vehicles. For example, there has been considerable effort at UC Davis aimed at development of a laser-based system to measure vehicle lengths. In low speed situations, a change in velocity while the vehicle is in the detection zone could yield an inaccurate calculation of vehicle length and hence would cause problems in identifying vehicles. To help alleviate this problem, reflectance signatures of vehicles in the IR might be used as a metric for identifying vehicles. The concept is to project light from two lasers of different wavelengths onto vehicles. The reflected light would be sensed, e.g., using a photodiode array. It is expected that the reflectance ratios of two different frequency lasers in the IR portion of the spectrum will be a function of the vehicle paint and therefore provide a consistent metric.

This report discusses measurements of vehicle paint reflectance characteristics. These measurements involved using equipment that was available at UC Davis. The measurements were performed by a group of four undergraduate students (Lucien Burgert, Sumit Harit, Thomas Houghton and Jonathan Lee) in collaboration with Prof. B. D. Shaw. Measurements were taken on different-colored vehicles parked in a UC Davis parking lot.

6.1 Measurement System

The measurements were made using two portable spectrometer systems available at UC Davis. The first system was an Ocean Optics S1000 dual-channel UV-VIS and VIS-NIR spectrometer. This spectrometer was interfaced with a laptop computer using a National Instruments DAQ 700 PCMCIA card. The second spectrometer was an Ocean Optics S2000 VIS-NIR temperature-regulated spectrometer. This spectrometer was also interfaced with a laptop computer, though in this case a USB connection was used. The S2000 spectrometer was used because the S1000 spectrometer suddenly stopped working during the course of taking measurements. The S1000 spectrometer was from Prof. B. D. Shaw's lab while the S2000 spectrometer was borrowed from Prof. David Slaughter's lab in the Biological and Agricultural Engineering Department of UC Davis. The data presented in this report were obtained using the S2000 spectrometer. The spectrometers could measure spectral reflectance data over the approximate range 650 nm to 1100 nm.

Both spectrometers employed an Ocean Optics R200-7 fiber-optic reflection probe coupled with an Ocean Optics LS-1 tungsten-halogen light source. With the reflection probe, light from the incandescent light source was directed down an optical fiber and onto the surface of a vehicle. The reflected light was collected using other optical fibers that were just next to and aligned with the first fiber. This collected light was directed to the spectrometer. The spectrometer measured the intensity of the reflected light as a function of wavelength, providing data that could be used to calculate spectral reflectance characteristics.

Two different types of reflectance measurements were taken, specular and diffuse. The diffuse measurements were taken at a 45-degree angle of incidence and the specular measurements were taken such that the incident light from the reflection probe was directed in a perpendicular fashion onto the vehicle paint. The reflectance probe was mounted in a special metal housing (an Ocean Optics RPH-1 Probe Holder) with black foam taped to the bottom, shielding the measurement zone from ambient light. The probe holder, which was painted black, had two holes that the reflectance probe could be inserted into. One hole was at a 45-degree angle relative to the vehicle surface and the other was normal to the vehicle surface. The reflectance probe was mounted in the appropriate hole for either the diffuse or specular measurements.

Each spectrometer was calibrated using two different types of calibration surfaces. For diffuse measurements, a diffuse white surface (i.e., a WS-1 Diffuse Reflection Standard obtained from Ocean Optics) with essentially greater than 98% reflectance over the wavelength range of interest was employed. For specular measurements, a first-surface aluminum mirror was used as the calibration standard, providing a reference for specular reflection characteristics. This mirror is expected to have about an 80% to 85% reflectance over the wavelength range of interest. Calibration data indicate that the first-surface mirror has a constant reflectivity over the spectral range 750 to 1000 nm. These calibration data were obtained by comparison of first-surface mirror reflectance data with reflectance data obtained using the diffuse white surface.

During an experiment, the spectrometer that was used was coupled to a laptop computer. This computer provided control of the spectrometer as well as data storage. The entire assembly was placed on a wheeled cart for easy movement. Two uninterruptible power supplies were used to supply power during the experiments. These power supplies were also placed on the cart.

6.2 Reflectance Measurements

6.2.1 Diffuse Measurements at a 45-Degree Angle Relative to the Painted Surface

Diffuse reflectance measurements were made from about 800 nm to about 1100 nm. The measurements indicated that overall IR vehicle paint reflectance levels could vary significantly with

the type of paint (i.e., visible color). However, for some paints (e.g., white paints), IR reflectance levels sometimes varied only by small amounts (a few percent or less) over the range 850 nm to about 1000 nm, that is, vehicle paint reflectance did not vary strongly with wavelength over this portion of the spectrum. For other paints (e.g., gold paints), larger changes were apparent. Data from these measurements are shown in Appendix E.

6.2.2 Specular Measurements Perpendicular to the Painted Surface

Because the first-surface mirror provided uniform reflectance over the range 750 to 1000 nm, specular reflectance measurements were made from about 800 nm to 1000 nm. The measurements indicated that overall IR vehicle paint reflectance levels could vary significantly with the type of paint (i.e., visible color). However, for a given paint, vehicle paint reflectance generally did not vary strongly with wavelength over this portion of the spectrum, though some variations were present. Data from these measurements are shown in Appendix F.

6.2.3 Reflectance Ratios

Table 5 shows reflectance ratios calculated using the reflectance data in Appendices A and B. These ratios were calculated using data at wavelengths of about 850 nm, 1000 nm and 1100 nm. In this table, R_1 is the ratio of reflectance at 1000 nm to the reflectance at 850 nm and R_2 is the ratio of reflectance at 1100 nm to the reflectance at 850 nm. Data for R_2 are not shown for specular reflectance because the first-surface mirror did not exhibit uniform reflectivity for wavelengths outside the range 750 to 1000 nm. In addition, table entries with dashes correspond to situations where a particular measurement was not taken (e.g., because a car left the parking lot before all measurements could be obtained). The data in Table 1 show that significant variations in reflectance ratios can exist between vehicles, depending on the vehicle paint, the wavelength ratio and whether the measurement is diffuse or specular.

Vehicle	**R_1 (Diffuse)**	**R_2 (Diffuse)**	**R_1 (Specular)**
Green Subaru Outback	1.62	4.57	0.54
Green Audi A4	-	-	0.19
Dark Green Honda CRV	2.17	10.0	0.05
Gold/Silver Buick Park Ave.	1.56	3.12	-
Gold Mazda Protégé	1.95	7.74	0.64
Gold Toyota Camry	1.71	4.17	0.87
Gold Buick LeSabre	-	-	0.77
White Ford Taurus	0.96	1.06	0.45
White Chevy Astro Van	0.93	0.74	0.33
White Honda Accord	0.97	1.14	0.44
Metallic Silver LeSabre	1.45	2.51	-
Silver Honda Accord	1.73	4.90	0.77
Silver Buick LeSabre	-	-	0.59
Royal Blue Nissan Altima	0.80	0.81	-
Blue Ford Ranger	1.37	1.23	0.79
Red Toyota Celica	0.93	0.77	0.21
Black Ford F150	-	-	0

Table 5: Reflectance ratios calculated from the data in Appendices E and F.

6.3 Discussion

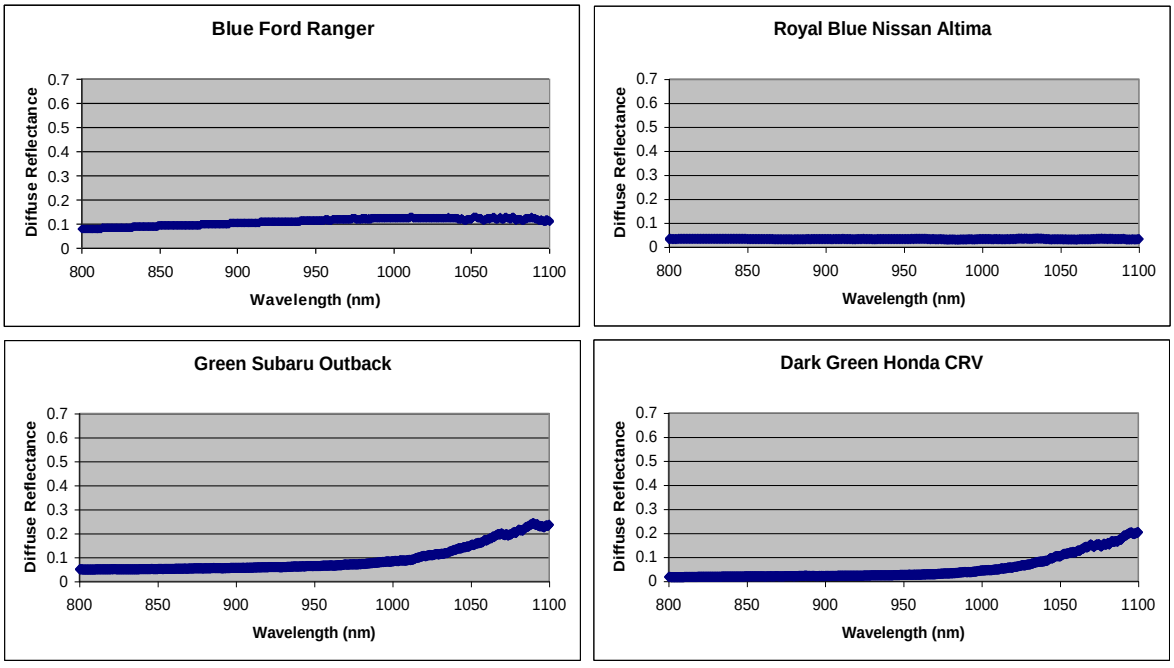
The results on diffuse and specular reflectance suggest that a laser-based measurement system that measures reflectance ratios of two different frequency lasers in the IR portion of the spectrum could be successfully developed. However, the system will likely have the greatest chance of success if

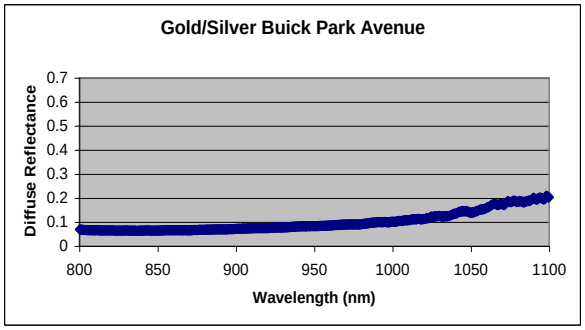
most of the reflectance data obtained is for diffuse reflectance. Because reflectance characteristics can vary somewhat slowly with wavelength, a key to the successful development of this type of system will be to employ lasers that have significantly different wavelengths. Since vehicle surfaces are often curved, a reflectance system that employs lasers operating in a vertical plane would likely obtain mostly diffuse reflectance data, though the angles of incidence on a vehicle might not differ significantly from the vehicle surface normal. Alternatively, the laser reflectance system might be operated at an angle (relative to vertical). For optimization of this type of system, it would be worthwhile to perform further diffuse reflection measurements at different angles of incidence to determine whether an optimum angle exists where spectral reflectance characteristics are strongest.

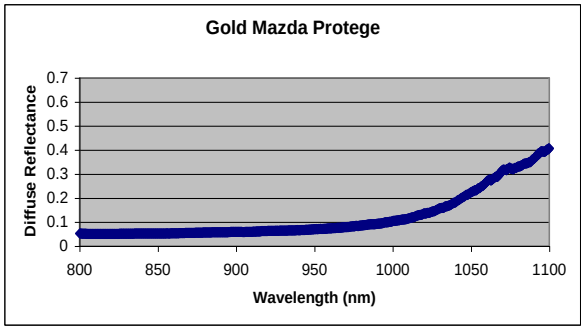
It is noted that since the data in Table 5 and sections 6.4 and 6.5 indicate that spectral reflectance characteristics can vary with angle of incidence, a laser-based reflectance system would need to employ methodology that accounts for variations in angle. For example, the system could measure the reflectance ratio over an entire vehicle and compute an average. Alternatively, the peak reflectance ratio measured from a vehicle could be used. The optimum approach will likely be determined by experimentation with an actual laser system under highway conditions.

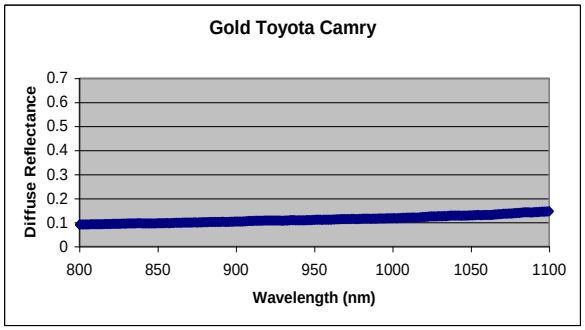
Finally, it would be of interest to extend these measurements to other angles of incidence as well as further into the IR (i.e., longer wavelengths). Characterizing spectral reflection characteristics at different angles may provide useful data that could be used to optimize a laser-based reflectance system. In addition, going further into the IR should allow for greater differences between laser wavelengths, which may allow for larger reflectance ratios and better differentiation between vehicles that are almost the same color (e.g., different shades of white).

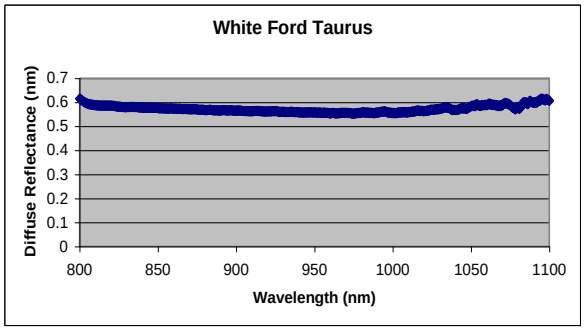
6.4 Diffuse Reflectance Data

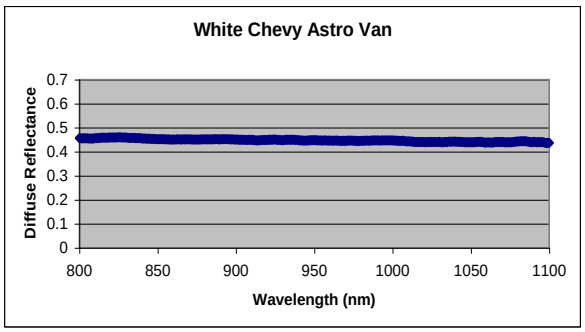


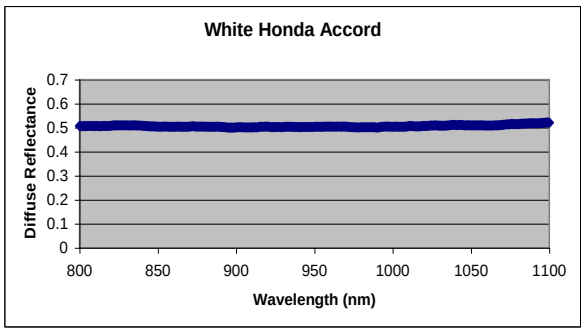

Gold/Silver Buick Park Avenue
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

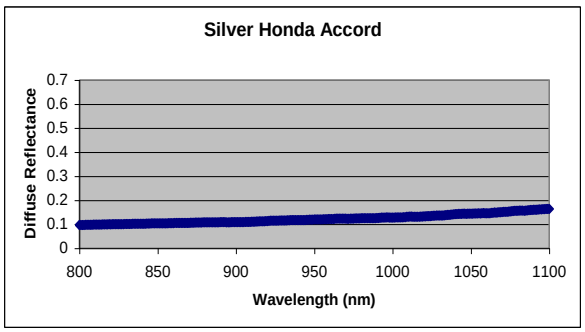

Gold Mazda Protege
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

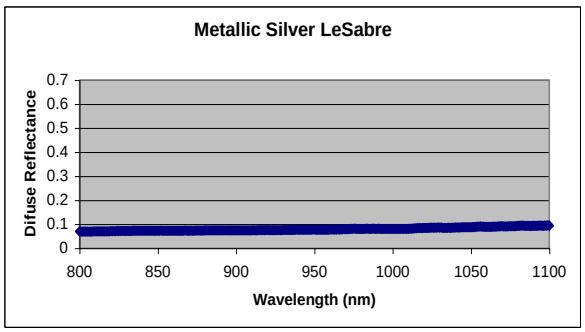

Gold Toyota Camry
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

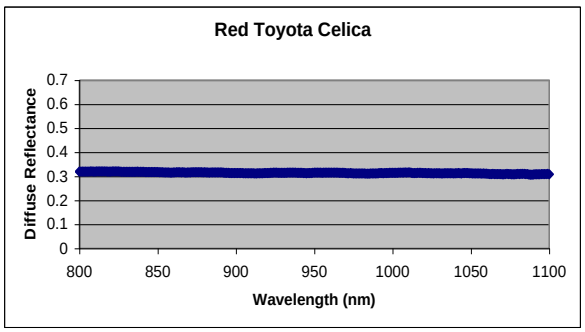

White Ford Taurus
Diffuse Reflectance (nm)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)


White Chevy Astro Van
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

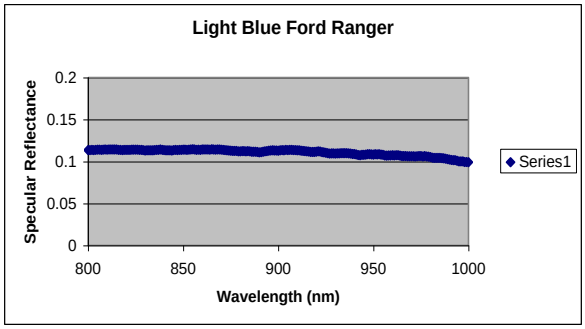

White Honda Accord
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

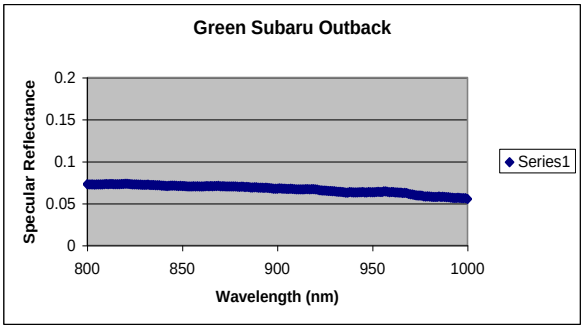

Silver Honda Accord
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

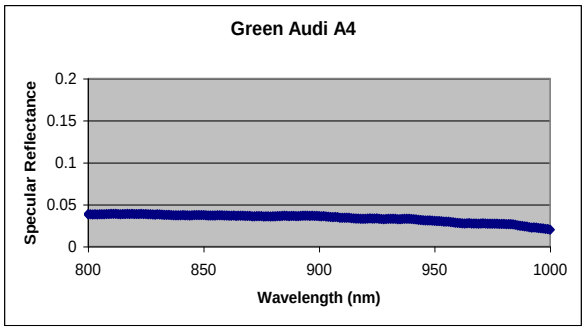

Metallic Silver LeSabre
Difuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

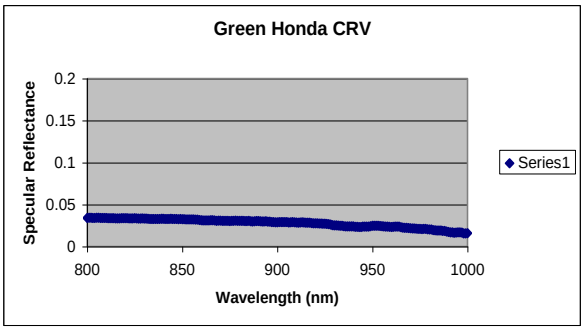

Red Toyota Celica
Diffuse Reflectance
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
800
850
900
950
1000
1050
1100
Wavelength (nm)

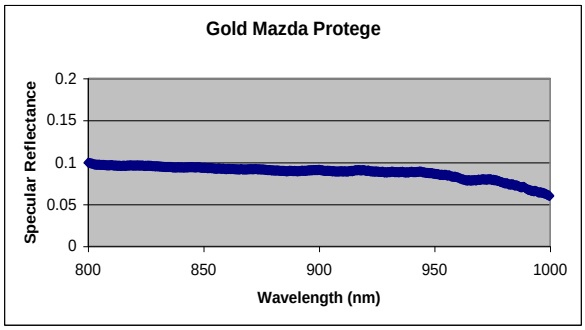
6.5 Specular Reflectance Data

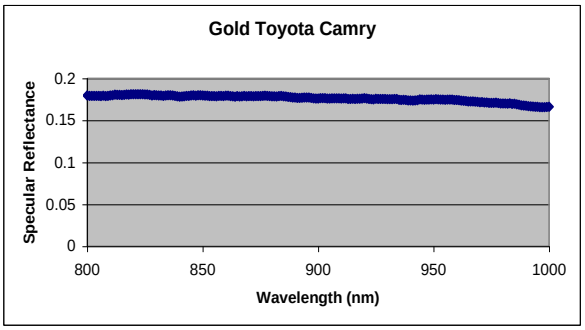


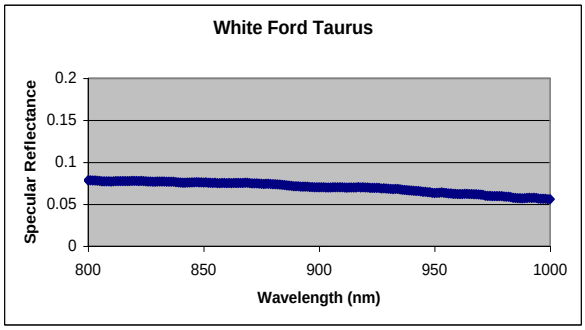


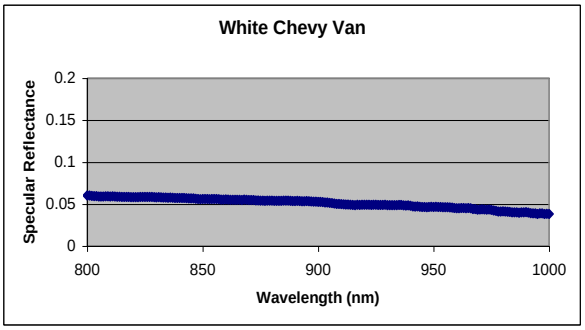


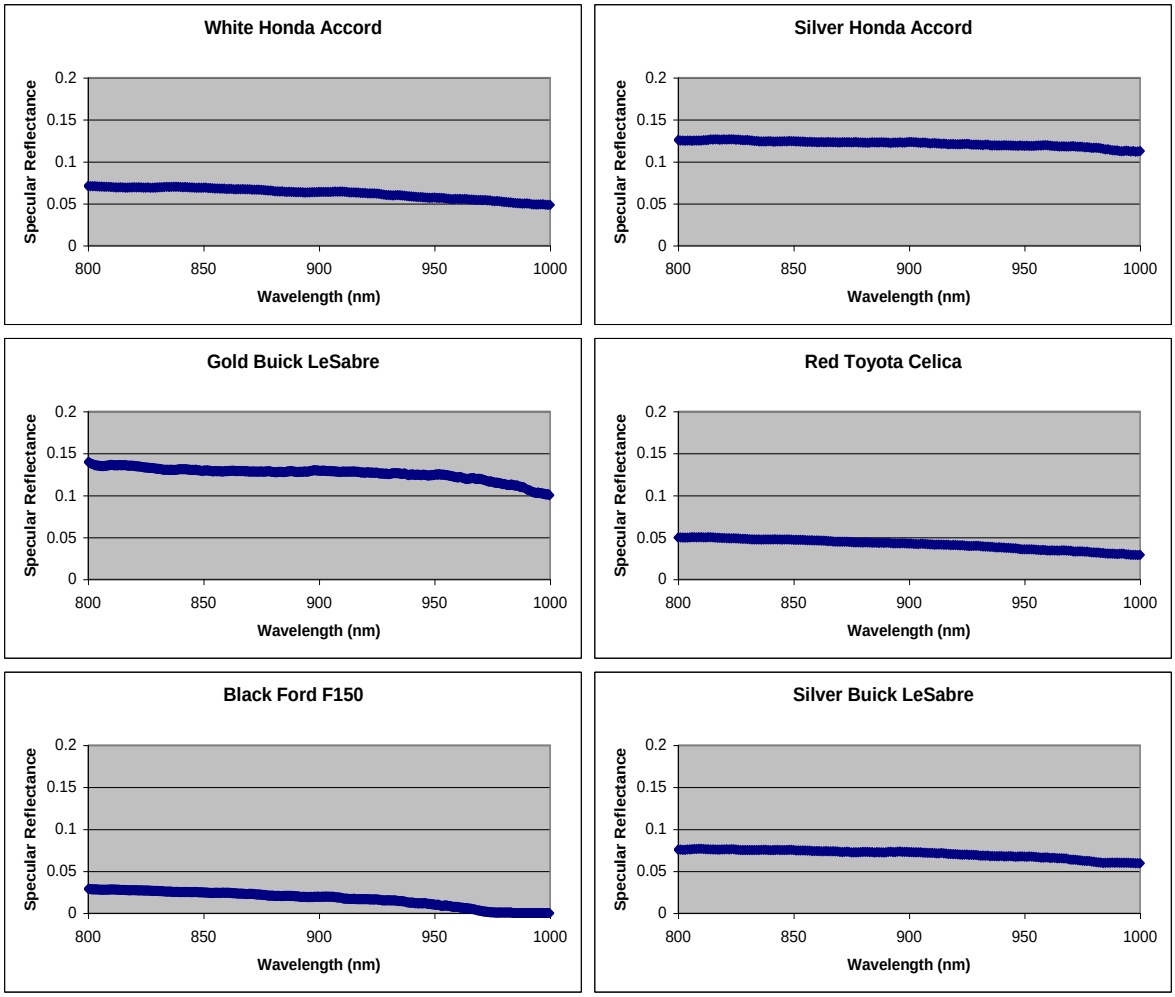












7 Future Work

7.1 Networking

There are two possible network configurations for the laser system. The laser system can either be connected to an existing network through an Ethernet cable via the Rabbit processor or through the use of wireless technology. Utilizing the wireless network technology is more favorable since the laser systems will be mounted above high ways and hooking them up to a network through a hard-line would require routing miles of Ethernet cable for just one system. Figure 7-1, on the following page, shows a network configuration utilizing wireless technology from Cisco.

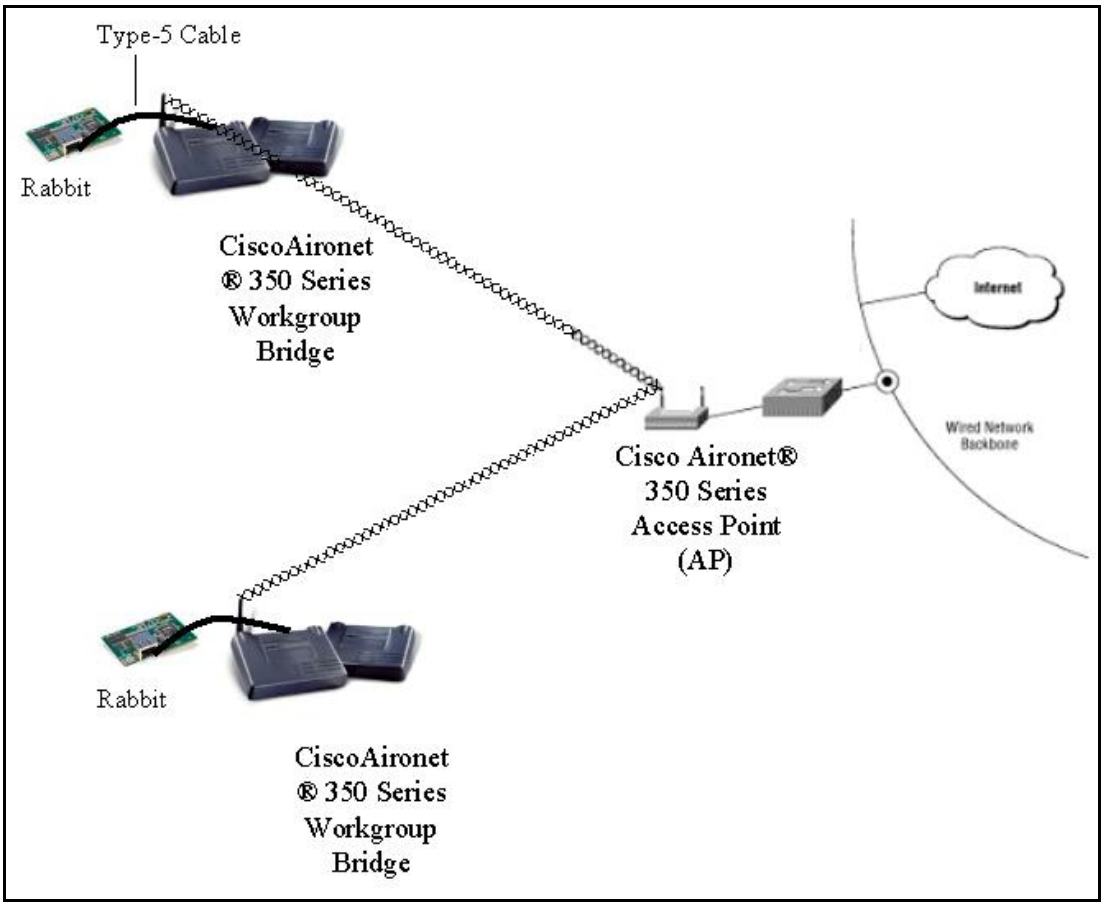


Figure 7-1: Possible Network Configuration

In the example network configuration, we have the Rabbit Processor, which provides the laser system with a 10/100Mbps Ethernet adapter, connected to a Cisco Aironet Bridge that wireless connects to a Cisco Aironet Access Point. The Access Point is connected to a local network through a switch or hub.

Listed in the table below are the required equipment needed to setup up the example network configuration specified in the previous section.

Product name	Qty.
Cisco Aironet 350 workgroup bridge	4
Cisco Aironet 350 Access Point	2
Cisco 1538 Series Micro Hub 10/100	1

Table 6: Equipment Requirements for Network Configuration

7.2 USB DIO96 Board

The current LBDS utilizes a NI DIO board to record the output of the microchips. The NI DIO board must be plugged into a computer PCI slot in order to work. This means that in order to test the system, a computer must be taken with it. Lugging around a minitower, a monitor, and any required

accessories becomes cumbersome especially when roof testing or field testing the system. Instead of utilizing the NI DIO PCI card, a laptop with a USB DIO96 board will be used. This reduces the amount of setup time and travel weight.

7.3 NI DAQ PCMCIA Card

Aligning the signal onto the APD array is rather complicated and very important. Any small deviations cause degradation in the signal strength reducing the overall performance and ability of the LBDS. In order to simplify the aligning process and decrease the setup time, a NI DAQ PCMCIA card will be utilized to determine the rotation of the reflected laser beam. Currently, each output signal from the APD array is checked separately to determine if the signal strength is acceptable. Since there are 8 output signals, this process becomes tedious and with ideas of expanding the system to 24 output signals, the process would take even longer. The NI DAQ PCMCIA card will be used to show the output of all the signals at once on a laptop computer. This will reduce the setup time required and improve the precision of the overall system.

7.4 Signal Expansion

The current LBDS uses only 8 output signals from the APD array even though there are 25 available output signals. The next step is to expand the system to utilize 20 output signals which will improve the resolution of the system. Increasing the resolution of the system will allow us to obtain a more defined profile of the vehicle that passes under the system. This entails a more accurate ability to recognize the vehicle for travel time calculations.

7.5 Optical Design

The current optical system is high inefficient. It takes too much time and steps to align the optical system. It has been found that any small deviation of the lens systems reduces the overall performance of the LBDS substantially. A new optical design will be created and researched in order to minimize the amount of steps and time required to align the optical system. This will increase the overall performance and allow for more actual testing time instead of wasting the time aligning the system.

7.6 Mechanical Damping

The original design incorporated two lasers where this design only implements one laser. The bracket had to meet certain criteria of having a 1kg maximum total weight including the lasers, be able to adjust the yaw angle of the top laser, and have 20^{o} of horizontal tilting freedom for each.

Figure 7-2(a) shows the laser bracket from the initial laser system before any modifications.

Figure 7-2(b) shows the new laser bracket created for the LBDS. Figure 7-3 shows the new proposed design for the single laser bracket. There are certain shafts that have multi-material centers that allow for better damping of vibrations along the shaft.

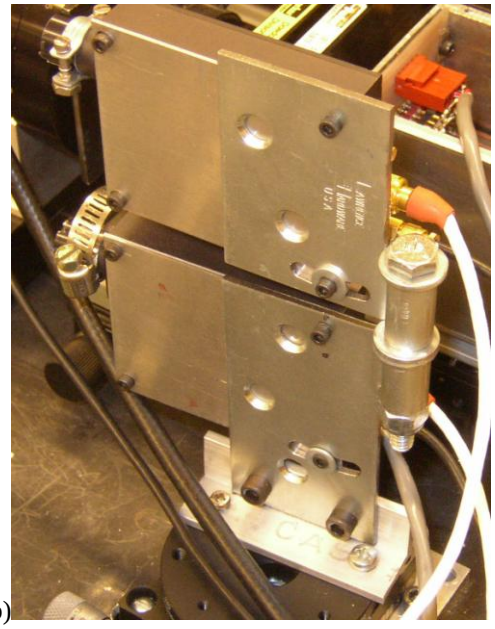
Using web based CH, a plot of the deflection and shear of the mounting shaft (standard stock diameter 20mm and length 152mm) with a 200g mass located at the end is shown in Figure 7-4 and Figure 7-5. It can be inferred that the deflection of the shaft is minimal such that the dimensions of the shaft are adequate for its specific use. The results from CH give a reaction force at x=0.15mm of R_1=2N, a reaction moment of M_1=-0.28Nm and a maximum displacement of y_{max}=-3.65e^{-6}m.

The damping materials shown in Figure 7-6(a) and (b) were found from *www.wmberg.com* to have damping characteristics that would meet the needs of this system. The item in Figure 7-6(a) are damping bushings for the locations where the system would mount to the trolley attached to the freeway. The item in Figure 7-6(b) is the isolation pads to the individual components within the system. The specifications for these damping materials can be found in the appendix of this report. It can be seen that these damping materials are capable of damping out frequencies of 20Hz up to 75Hz.

In the event that a larger shock occurs that the bushings cannot absorb, a possible solution here is to have the trolley absorb that force. The implementation of these components in the laser system is shown in Figure 7-7 thru Figure 7-9. Potential downfall of this design would occur when a shock to the system is larger than the allowable damping of the bushing. In such an event the residual shock that is not absorbed would transfer to the components within.

Having the laser system mounted to the trolley with an inner tube as the damping mechanism between them is another solution to this problem. This inner tube can act as both a damper and as a spring, depending on the air pressure of the tube. The air spring is shown in Figure 7-10(a) and the implementation of it is shown in Figure 7-11 thru Figure 7-13. The success of this design is heavily dependent on the material and manufacturing of the inner tube. Some potential problems related with this design are the result from prolonged interaction of the inner tube with the environment. Heat and cold weather would cause the air in the inner tube to expand and compress. Over time the air may leak out of the inner tube rendering it useless.

The electrical system will be housed in a box shown in Figure 7-14and Figure 7-15. The motherboard will be able to incorporate up to five digital amplifier PCBS and a RABBIT microprocessor. The box will also house the power supply for the laser system, which sits in front of the motherboard.

(a)  (b) 

(a) Original Laser Bracket. (b) New Laser Bracket

Figure 7-2: Laser Bracket Design

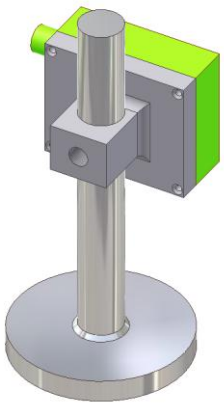


Figure 7-3: Proposed Laser Bracket

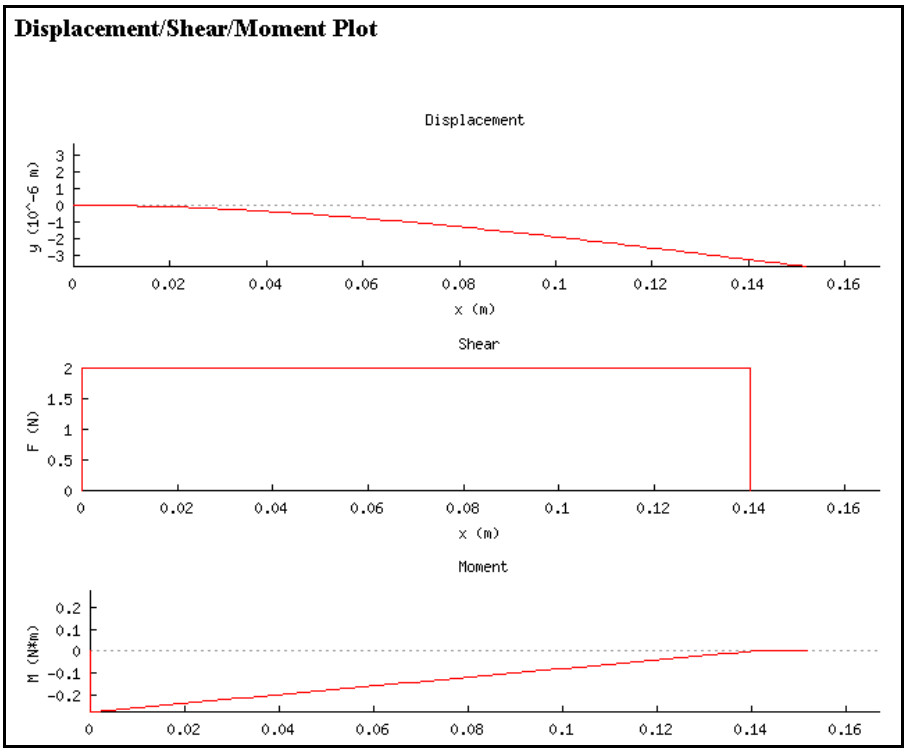


Figure 7-4: Deflection/Shear Plot for the Mounting Shaft

http://dragon.engr.ucdavis.edu/chhtml/toolkit/mechdesign/cantilever_conc.html

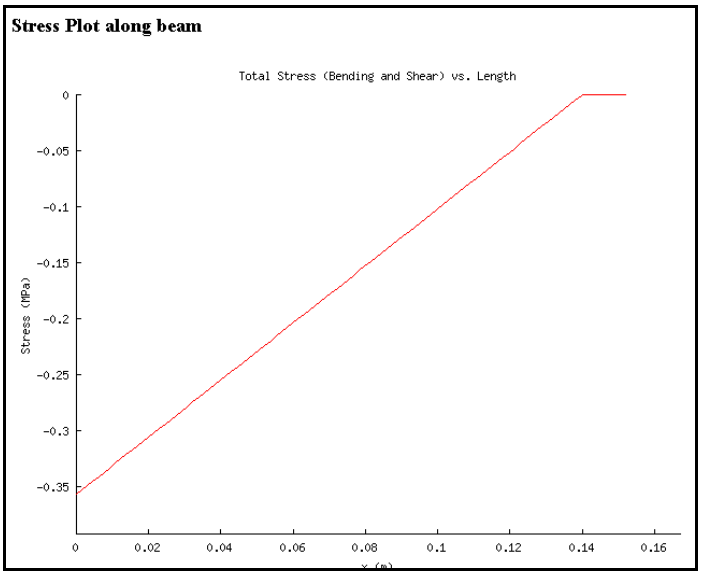


Figure 7-5: Stress Plot along the Mounting Shaft

http://dragon.engr.ucdavis.edu/chhtml/toolkit/mechdesign/cantilever_conc.html

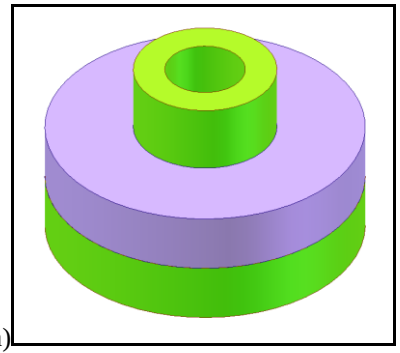
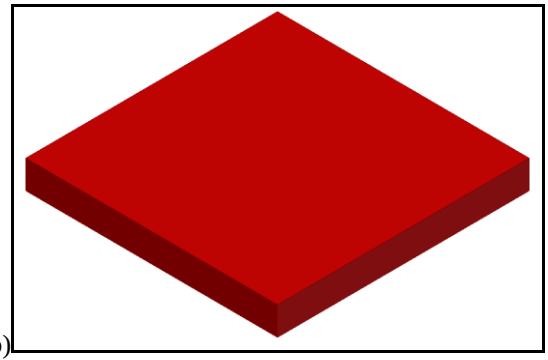
(a)  (b) 

Figure 7-6: (a) Damping Bushings for the Bottom Mounts of the Laser System

(b) Damping pads for the individual components inside the laser system.

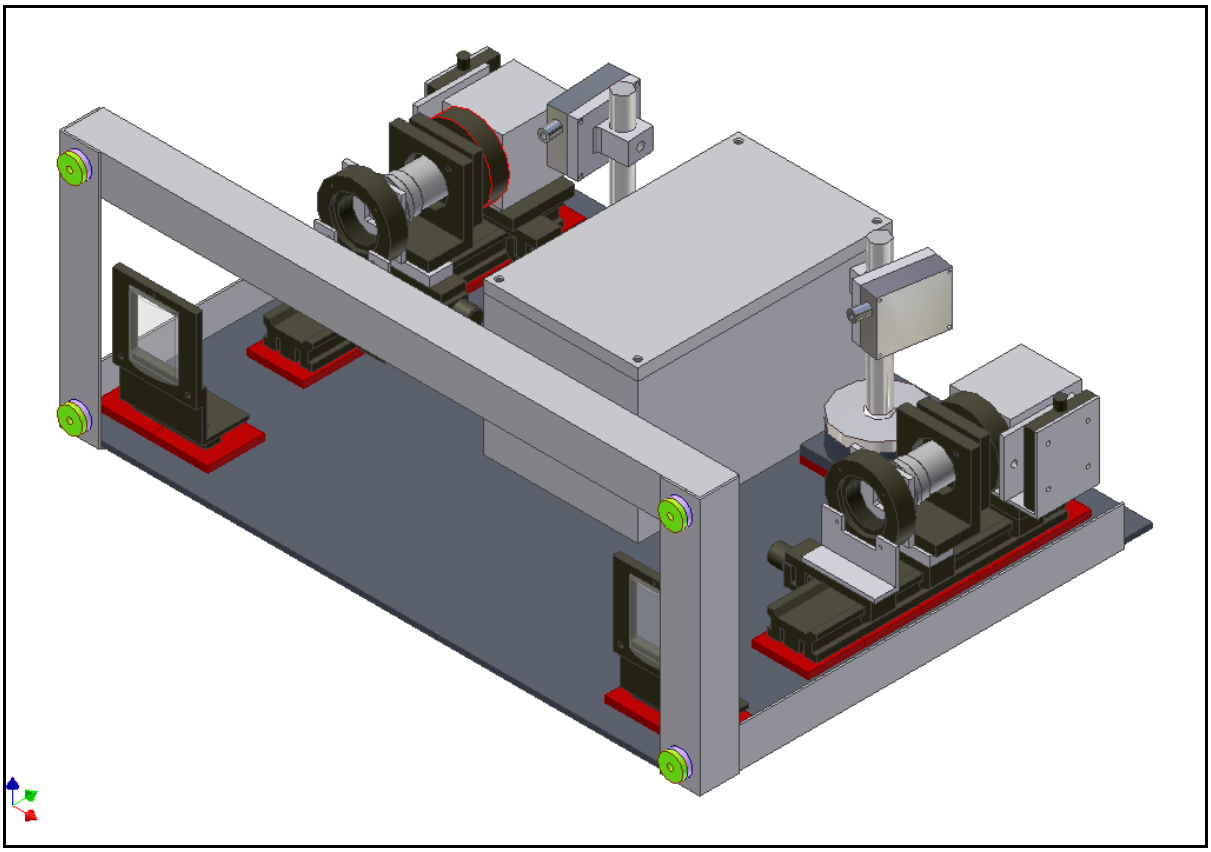


Figure 7-7: Isometric View of Laser System with Bushings

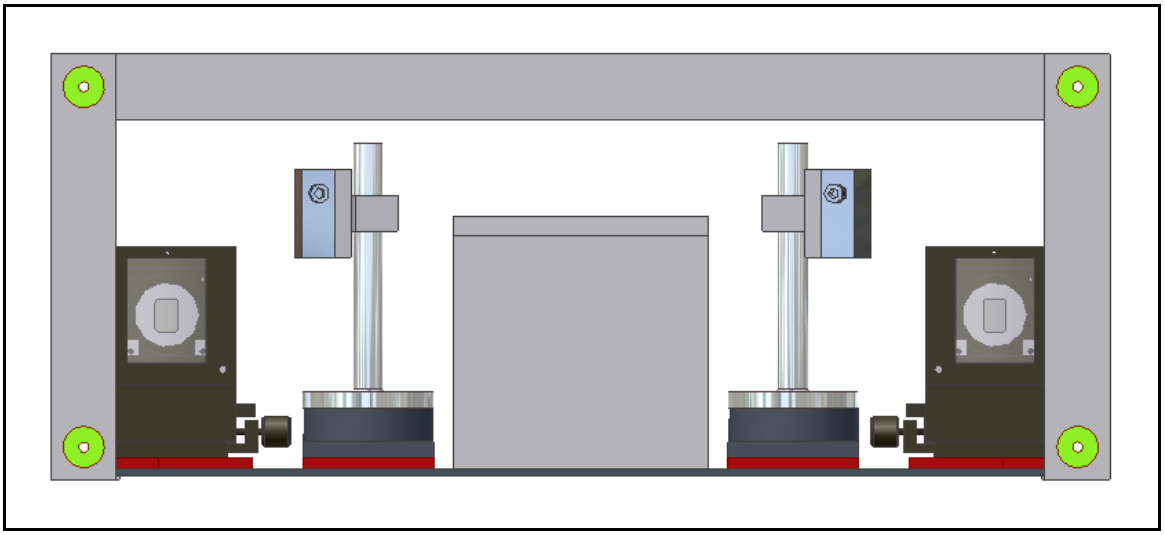


Figure 7-8: Front View of Laser System with Bushings

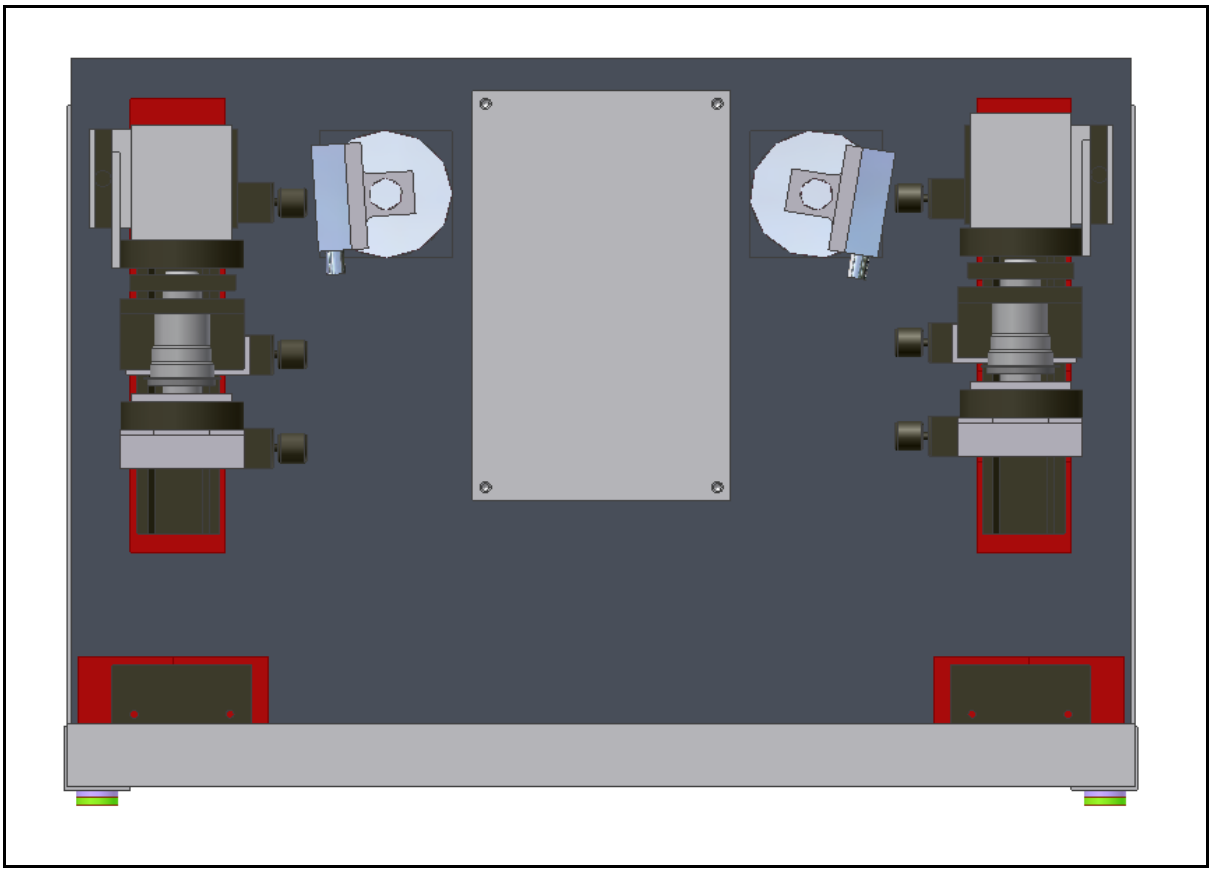


Figure 7-9: Top View of Laser System with Bushings

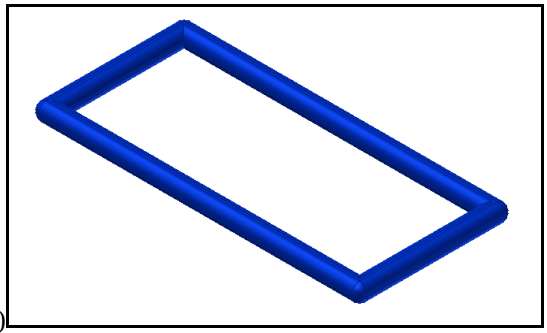
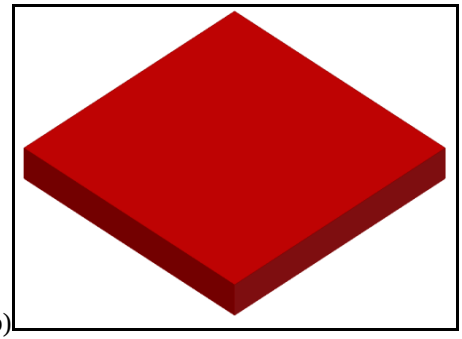
(a)  (b) 

Figure 7-10: Utilizing a Damping Spring

(a) Air spring for the bottom mounts of the laser system.

(b) Damping pads for the individual components inside the laser system.

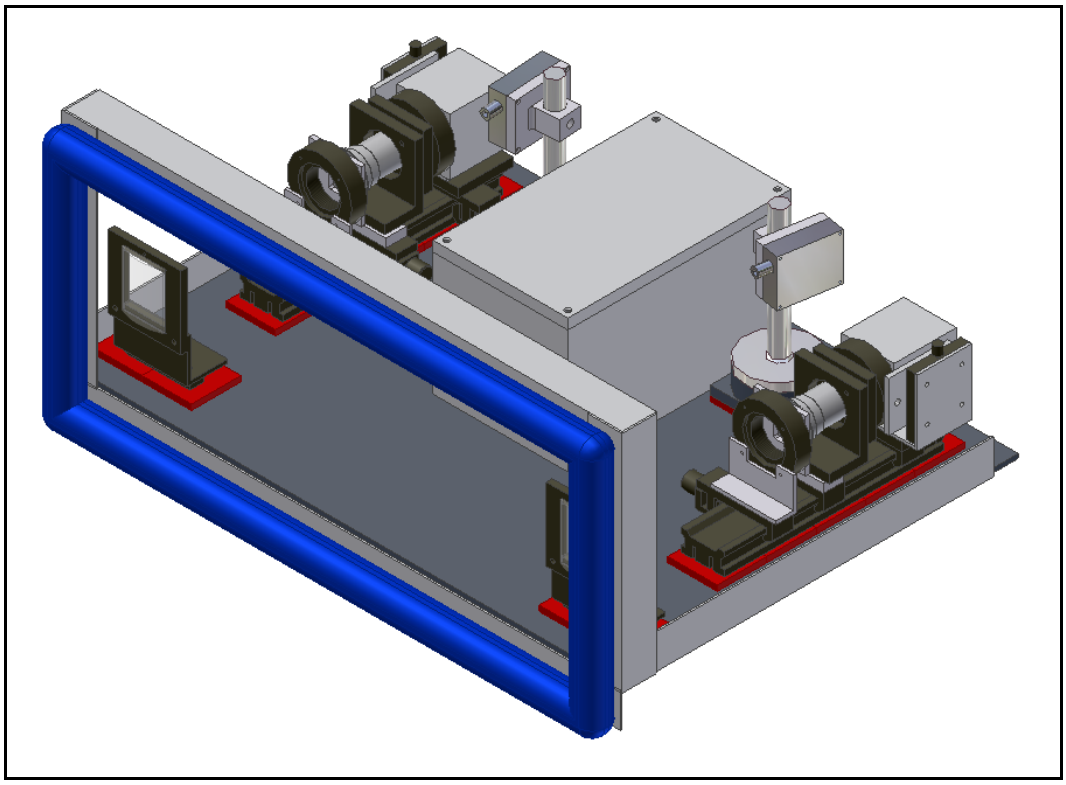


Figure 7-11: Isometric View of Laser System with Air Spring

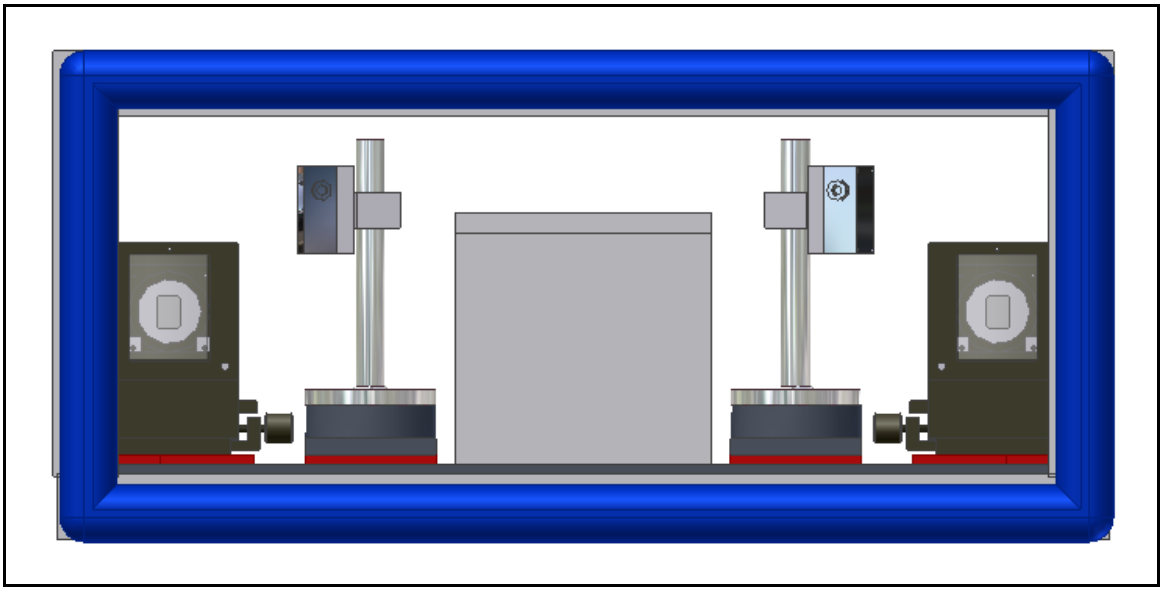


Figure 7-12: Front View of Laser System with Air Spring

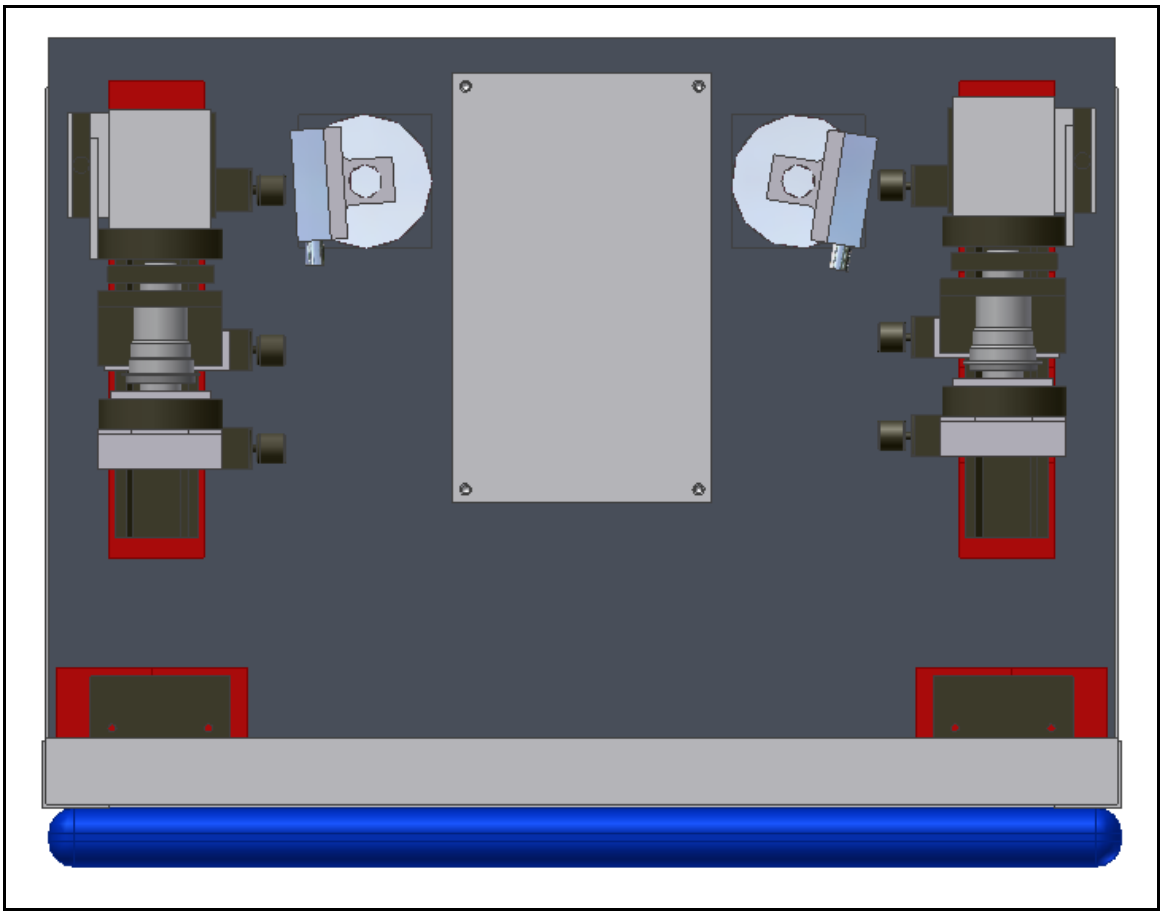


Figure 7-13: Top View of Laser System with Air Spring

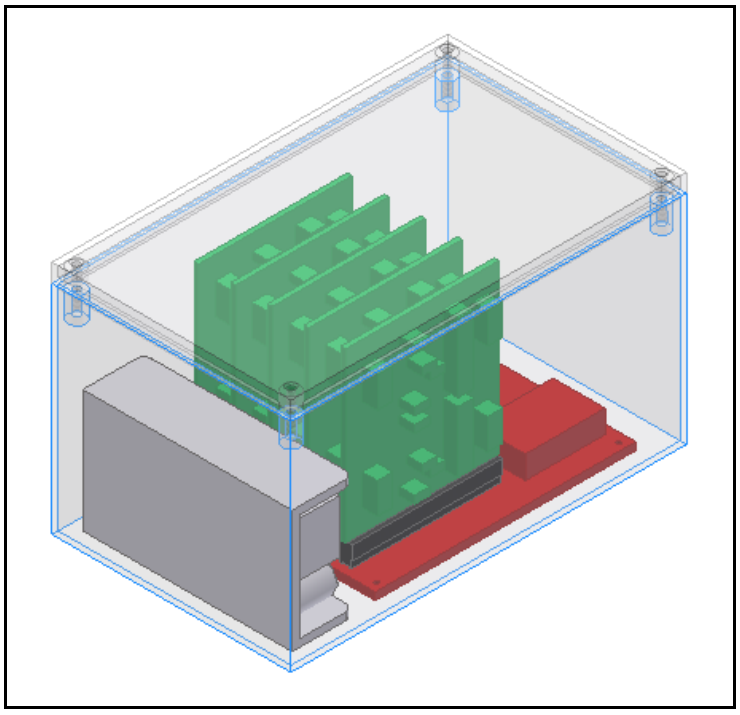


Figure 7-14: Isometric view of the electronics box with PCB and power supply.

The green PCB is the analog to digital amplifying converter, the red PCB is the motherboard interface, and the gray component is the power supply.

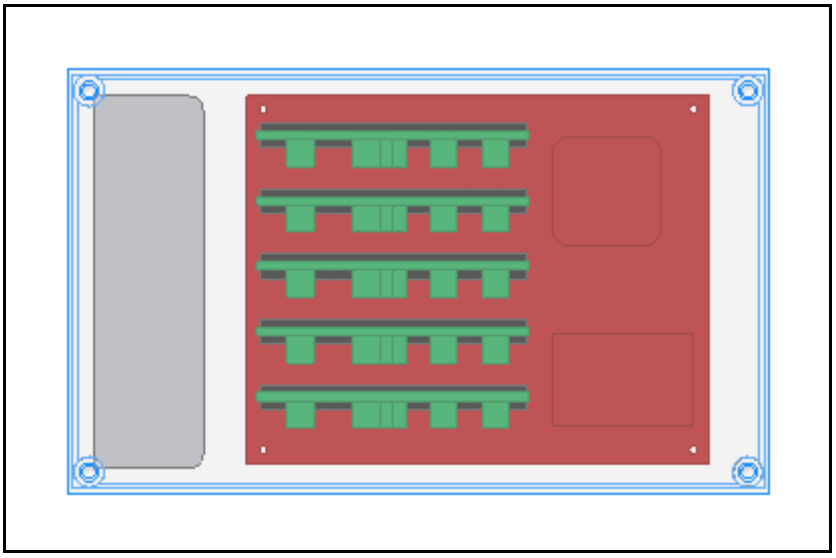


Figure 7-15: Top View of the Electronics Box with PCB Board and Power Supply

8 Conclusions

Working on the project of creating a real-time laser-based non-intrusive detection system for the measurement of true travel time on a highway, we have improved the system further. The control feedback loop was improved by adding a comparator. This helped reduce the noise on the input signal making the system more reliable. The power supplied to the lasers was separated in order to further reduce cross interference. New circuit boards were added with plug and play ability to give the system expandability and flexible in updating the software on the PIC microprocessors and debugging any unexpected problems. A new look on what the LBDS can do was done through tests in dealing with using optics to distinguish different vehicle colors and finding trace amount of pollutants in the air. Being able to distinguish color would give the LBDS a better ability to re-recognize a vehicle and further enhance its ability to calculate the true travel time of vehicles on a highway. Future works were considered in dealing with networking and reducing the amount of time taken during setup and aligning.

9 Acknowledgements

The financial support from CalTrans under the contract no. 65A0121 A01 is gratefully acknowledged. We would like to thank Matt Johnson at CalTrans and students at the Integration Engineering Laboratory at UC Davis for their assistance in field testing of LBDS on the highway. We are grateful for Maria Gonzalez's help in preparing the specifications in Appendix B. We also would like to thank Garfull Chan for his effort in the mechanical design of the LBSD. The efforts of Lucien Burgert, Sumit Harit, Thomas Houghton, Jonathan Lee in collecting the data for infrared reflectance of vehicle paints are gratefully appreciated.

10 References

[1] Palen, J, The Need for Surveillance in Intelligent Transportation Systems --- Part Two, *Intellimotion*, Vol 6, No. 2, 1997, pp. 1--17.

[2] Palen, J, The Need for Surveillance in Intelligent Transportation Systems, *Intellimotion*, Vol 6, No. 1, 1997, pp. 1--10.

[3] Ostland, M., et al., Simple Travel Time Estimation from Single-Trap Loop Detectors, *Intellimotion*, Vol. 6, No. 1, 1997, pp. 4--11.

[4] MacCarley, C. A., Advanced Imaging Techniques for Traffic Surveillance and Hazard Detection, *Intellimotion*, Vol. 6, No. 2, 1997, pp. 6--15.

[5] Malik, J., and Russell, S., Measuring Traffic Parameters Using Video Image Processing, *Intellimotion*, Vol. 6, No. 1, 1997, pp. 6--13.

[6] Coifman, B. "Using Dual Loop Speed Traps to Identify Detector Errors", *Transportation Research Record no. 1683*, Transportation Research Board, 1999, pp 47-58.

[7] Coifman, B., Beymer, D., McLauchlan, P., and Malik, J. "A Real-Time Computer Vision System for Vehicle Tracking and Traffic Surveillance", *Transportation Research: Part C*, vol. 6, no 4, 1998, pp. 271-288.

[8] Zhang, H. M. and Recker W. W. (1999) "On optimal freeway ramp control policies for traffic corridors", *Transportation Research,* B.33(6):471-436

[9] Zhang, H. M. (1999) "A mathematical theory of traffic flow hysteresis," *Transportation Research, B.*33(1):1-23

[10] Palen, J, ITS Roadway Detection Systems Development, personal communication, 1996.

[11] Coifman, B. A.,*Vehicle Reidentification and Travel Time Measurement Using Loop Detector Speed Traps*, Ph.D. Dissertation, University of California, Berkeley, 1998.

[12] Palen, J., Roadway Laser Detector Prototype Design Considerations, personal communication,1996.

[13] Tyburski, R.M., A Review of Road Sensor Technology for Monitoring Vehicle Traffic, *ITE Journal.* Vol. 59, no. 8 (Aug. 1989) Wangler, et al., *Intelligent Vehicle Highway System Sensor and Method*, U.S. Patent No. 5,546,188, 1996.

[14] Olson R. A.; Gustavson R L., Wangler R J., McConnell R E., Active Near-Field Object Sensor and Method Employing Object Classification Techniques, *U.S. Patent* No. 5,321,490, 1994.

[15] Wangler R. J., Gustavson R L., McConnell R E., Fowler K L., Intelligent Vehicle Highway System Sensor and Method, *U.S. Patent* No. 5,757,472, 1998.

[16] Wangler R. J., Gustavson R L., McConnell R E., Fowler K L., Intelligent Vehicle Highway System Multi-Lane Sensor and Method, *U.S. Patent* No. 5,793,491, 1998.

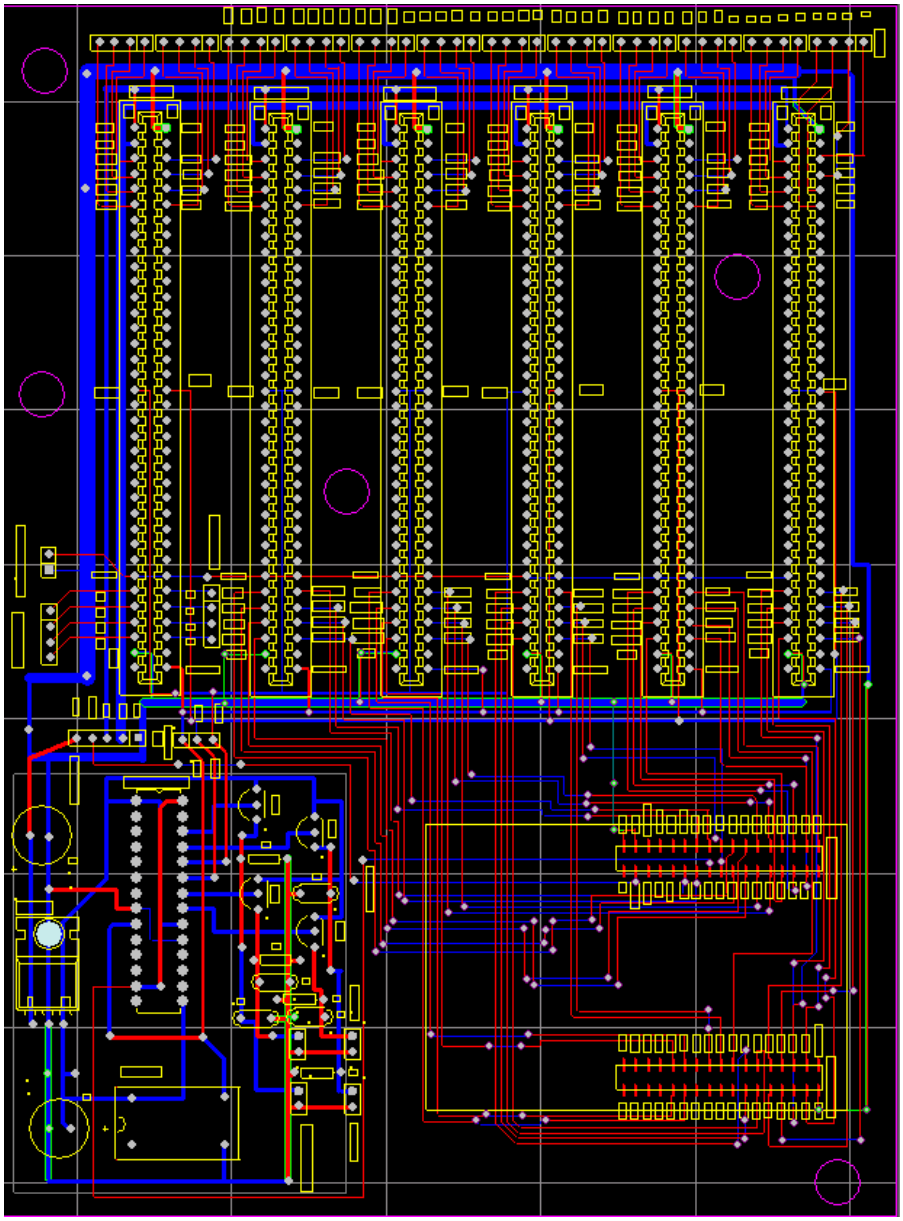
[17] Emerson, L., Mobil Video Surveillance and Ramp-Metering System, *Intellimotion*, Vol 6, No. 2, 1997, pp. 10--12.

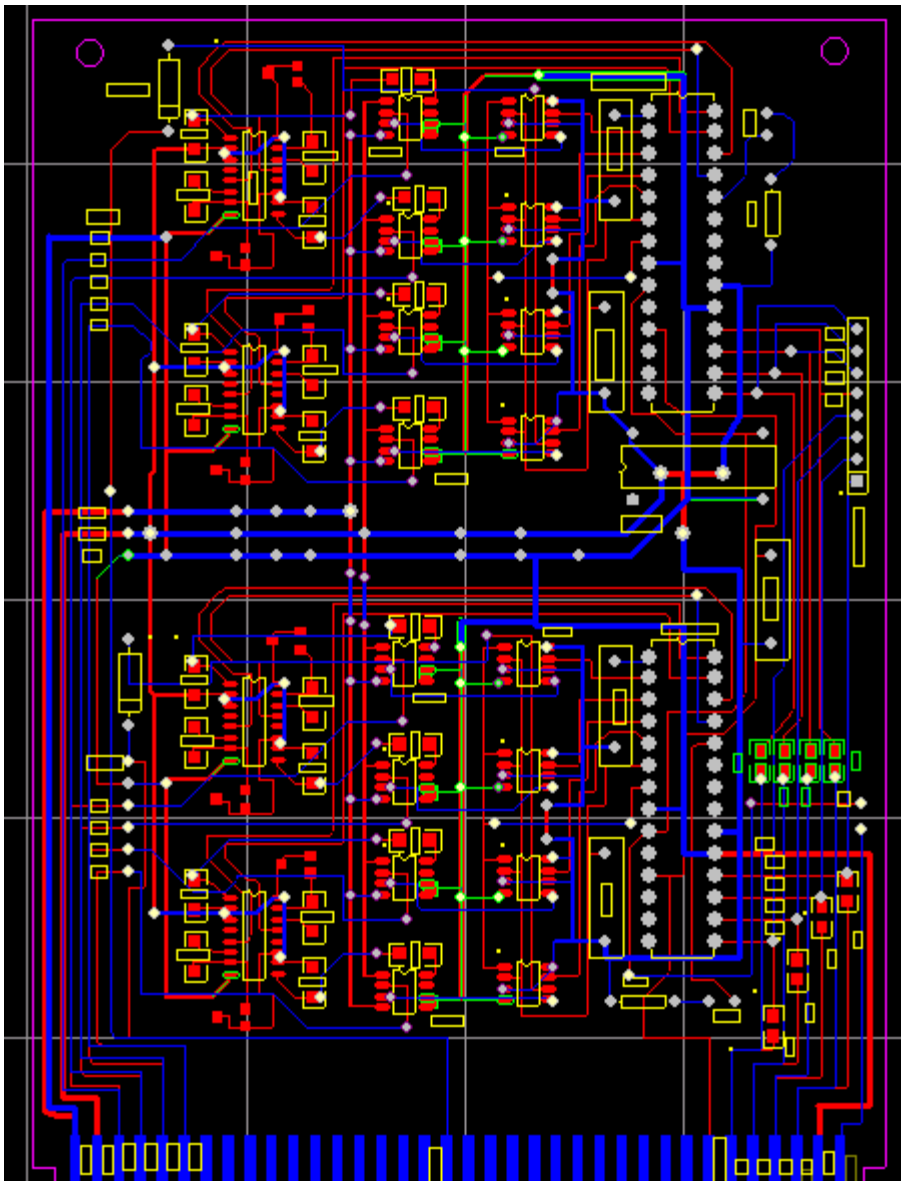
[18] Halvorson, G. A., *Automated Real-Time Dimension Measurement of Moving Vehicles Using Infrared Laser Rangefinders*, MS Thesis, University of Victoria, 1995.

[19] Cheng H. H., Shaw B. D., Palen J., Larson J. E., Hu X., Katwyk K. V., A Real-Time Laser-Based Detection System for Measurement of Delineations of Moving Vehicles, *CD-ROM Proc. of the ASME 19th Computers in Engineering Conference*, paper # DETC99/CIE-9072, Las Vegas, NV, September 12-15, 1999.

[20] Larson, J. E., Van Katwyk, K., Cheng, H. H., Shaw, B., and Palen, J., *A Real-Time Laser-Based Prototype Detection System for Measurement of Delineations of Moving Vehicles*, California PATH Working Paper, UCB-ITS-PWP-98-20. California PATH Program, Institute of Transportation Studies, University of California, Berkeley, September 1998.
[21] Bin Lin, Harry H. Cheng, Benjamin D. Shaw and Joe Palen, *"A Laser-Based Non-Intrusive Detection System for Real-Time Measurement of Delineations of Moving Vehicles on the Highway",* submitted to Optics and Lasers Engineering, ELSEVIER.
[22] *American National Standard for the Safe Use of Lasers*, Laser Institute of America, Orlando, 1986.
[23] LynxOS Lynx Real-Time Systems, Inc., *LynxOS Application User's Guide*, release 2.2.1, Los Gatos, CA, December 1993.
[24] Alessandro Brubini, Linux Device Drivers, 1st Edition, Oreilly, 1998.
[25] Rubini, Alessandro, *Linux Device Drivers*, First Edition, O'Reilly, 1998.
[26] Pedrotti, F. L. Pedrotti, L. S., *Introduction to Optics*, 2nd Ed., Prentice Hall, New Jersey, 1993.
[27] Ritchie, D. M. and Thompson, K. L., The Unix Time-Sharing System, *Commun. ACM*, vol.~17, No.~7, July 1974, pp.~365--375.
[28] Thompson, K., Unix Implementation, *The Bell System Technical Journal*, vol. 57, No. 6, July-August 1978, pp.~1931--1946.
[29] Kang, S. and S. Ritchie (1998). "Prediction of Short-Term Freeway Traffic Volume Using Recursive Least Squares and Lattice Filtering." Proceedings, 5th International Conference on Application of Advanced Technologies in Transportation, Newport Beach, USA. Edited by C. Hendrickson and S. Ritchie. American Society of Civil Engineers.
[30] Robert Siegel, John R. Howell. "Thermal Radiation Heat Transfer" 3rd Ed. 1992

Appendix A: Circuit and Board Diagrams





Appendix B: Specifications

Laser Specifications

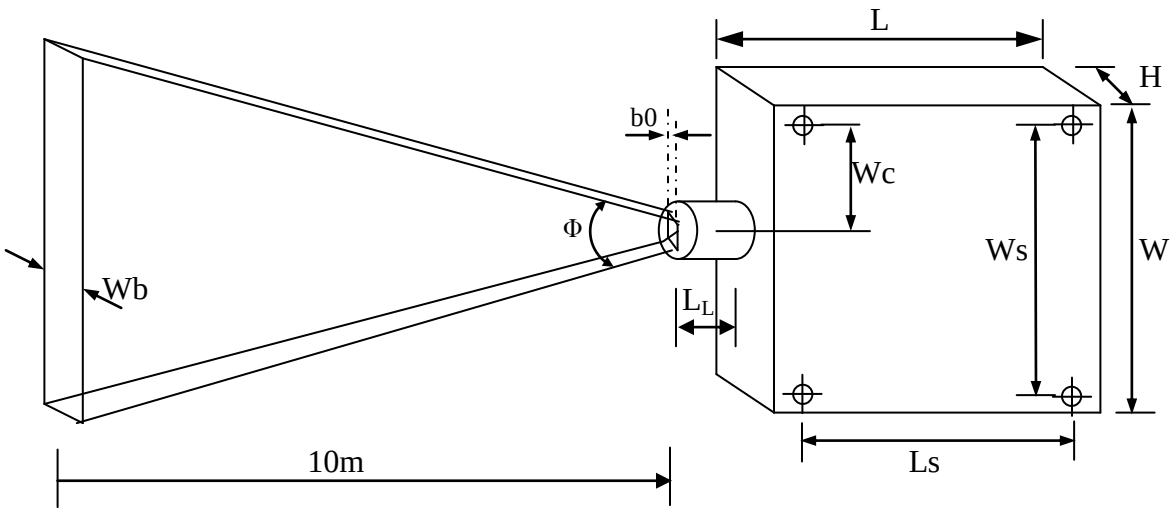
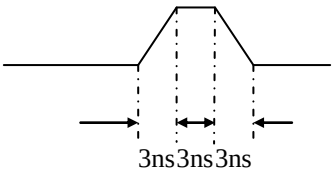


Figure 5:

This specification is for a repetitive pulse laser diode with fixed pulse duration.

Intrabeam Features:

- Center Wavelength: should be 905nm±2nm at 40˚C. When the temperature is lower than 40˚C, the laser module should be heated up (heating is easier to do than cooling). This should keep the center length of the laser in the range of 905 ± 2nm within variable environment temperatures.
- Pulse Repetition Frequency: adjustable up to 10KHz
- Pulse Width: fixed at 9ns ±0.1ns (no matter how long the trigger pulse is, the laser pulse width should be fixed at 9ns from the rising edge of the trigger pulse).



- Output Power: 60W (peak)
- Beam Shape: Rectangular.
- Major Axis Dimension : > 5mm (at the laser output lens)
- Major Axis Divergence Φ : 40˚ ±0.5˚
- Minor Axis Dimension b0 :7 ±0.5 mm
- Minor Axis Divergence : < 0.001˚
- Quantum Efficiency: >50% within ± 2nm of the center wavelength and >95% within± 8nm of the center wavelength
- Single beam: no laser beam outside of the described beam shape.
- The laser beam should be able to focus on a line at 5meters to 10 meters far away.

Power Supply

Input DC Power Voltage: Suitable for 5 to 12 V
Input Current (with DC input of 5V): <200mA
Input Trigger Pulse Voltage (TTL): 5V (up to 10KHz)

Temperature Features

Storage Temperature: -30˚C to +85˚C
Operating Temperature (full power): -20˚C to +55˚C
Center Wavelength at any Operating Temperature: 905nm ±10nm

Mechanical Dimensions

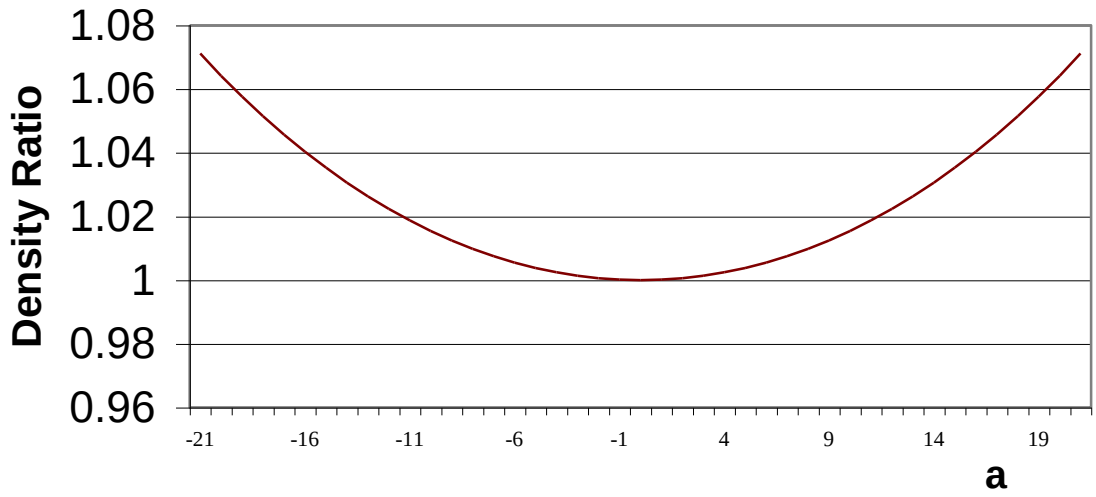
Our current laser module has four threaded thru holes. These thru holes are used to fix the laser to adjustable mounts. The position of these holes is arbitrary. We can machine our mounts accordingly although at least 3 holes are preferred. The length, width, and height of the module are also arbitrary, smaller sizes are preferred.

The Length of the lens, L_L, should be <20 mm.
Current Laser Module Dimensions:

Ws: 48mm
Ls: 61mm
Width W: 54 mm
Length L: 67 mm
Height H: 15 to 20mm

Laser Power Density Distribution

We require a laser power density distribution as shown in the figure below. Along the minor axis, the density distribution should be uniform. Ideally, the density distribution along the major axis should be proportional to $1/(\cos\alpha)^3$, where α is the angle of (major axis divergence Φ)/2 varying from -21˚ to 21˚. However, a uniform distribution along the major axis is also acceptable.



* Intrabeam parameters are critical for laser safety. If a parameter needs to be adjusted, we should be informed.

* After the laser diode is designed, we need all optical and electrical features with details for verification.

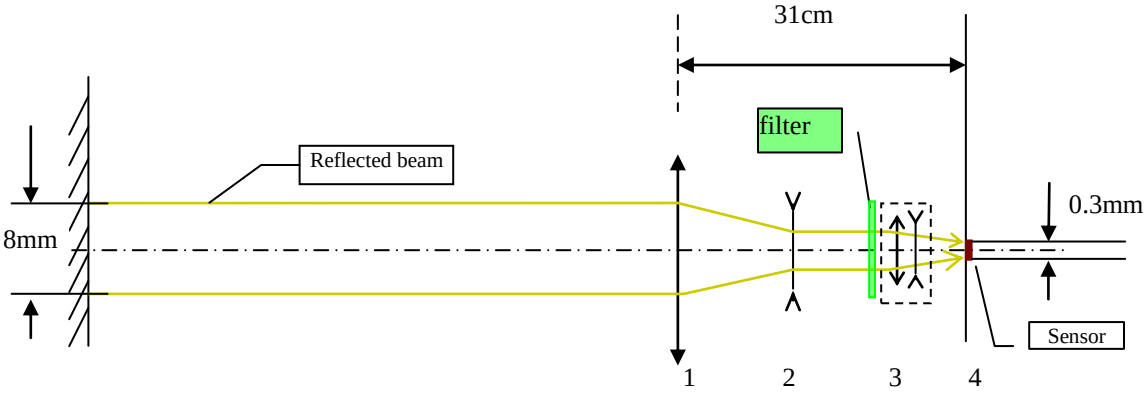
Optical Lens Specification

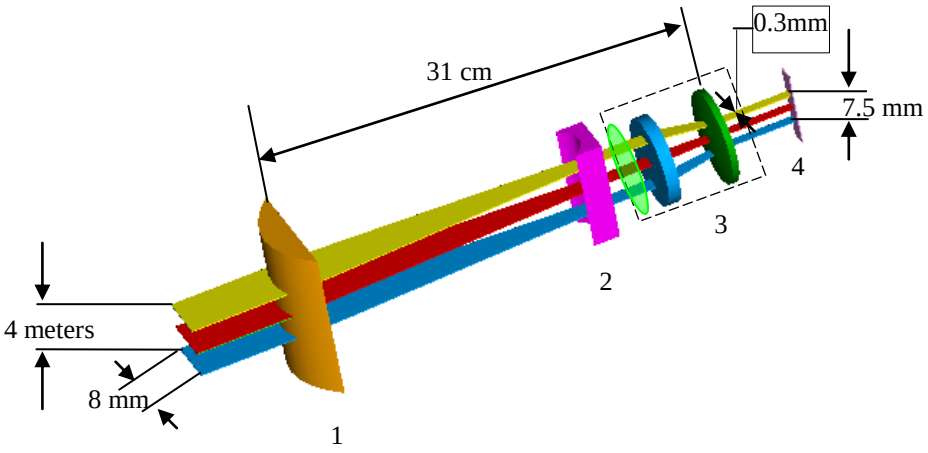
Requirements:

- The length of the image on the sensor should be 7.5 mm when the length of the laser line on the road surface is 4 meters. In addition, the width of the image on the sensor should include only the portion of the road that is illuminated by the laser. Assume that the range in distance between the road surface and lens 1 (marked below) is 5.4 to 9 meters.
- When the laser line with 8mm width on the ground shifts left or right 1.5cm or more, the image should go out of the active range of the sensor. The sensitive width of the sensor is 0.3 mm.
- The length of the optical system should be less than 31 cm.
- A band-pass filter (905 ±10nm) is needed to reduce solar noise. This filter should be replaceable so that other filters may be used.
- The whole optical system should be optically sealed.

One possible configuration:

One possible configuration is shown in the figures below.
The optical system is composed of 3 lenses.



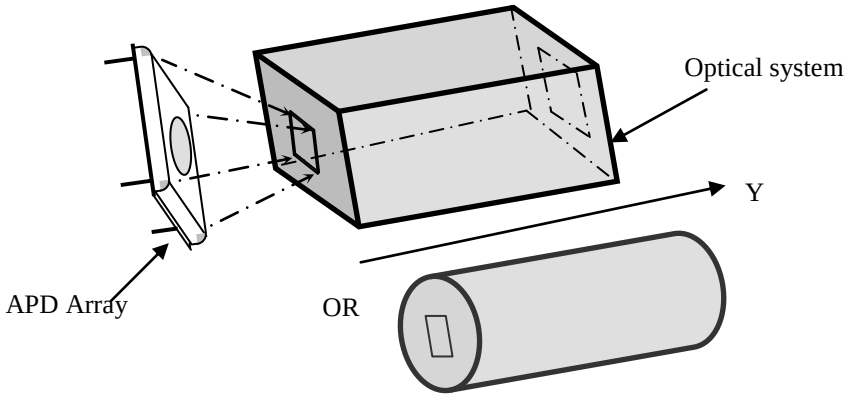


1. Positive cylindrical lens.
2. Negative cylindrical lens.
3. Imaging lens.
4. Sensor array.
5. A bandpass filter is located Between Len 3 and Len 2. It can be replaced manually

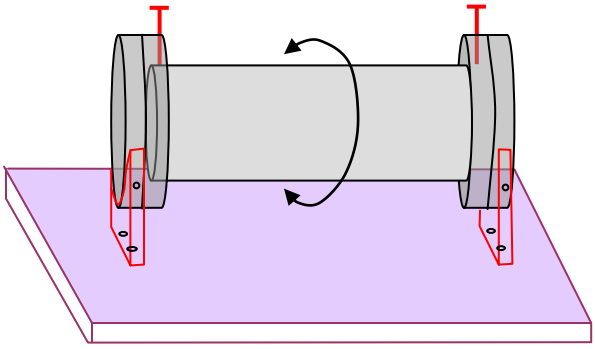
The specifications of the lenses are as follows:

Lens 1: Positive cylindrical lens. Smaller than 120mm*50mm
Lens 2: Negative cylindrical lens
Lens 3: f = 5 to 20mm (Manually Adjustable)

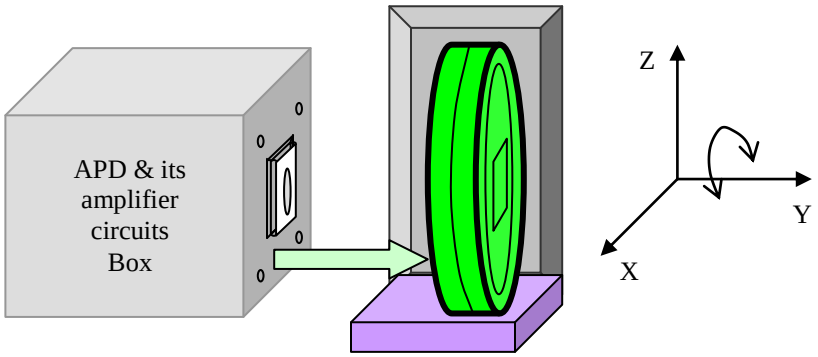
Assembly of APD Array and Optical system



1. The Figure above shows the optical system with the APD array. The optical system has two openings: one in the front, for the laser beam input, and one in the rear, for the ADP connection.
   - The junction between the optical system and the APD array should be completely sealed from unwanted light
   - The zoom and the aperture can be adjustable manually.
   - It should be able to focus on a line at a distance of 5 to 10 meters.
- The entire assembly should have a mount that allows rotation around the optical center as shown in the figure below. The optical system and APD array can be rotated together around Y axes about (30 degree).

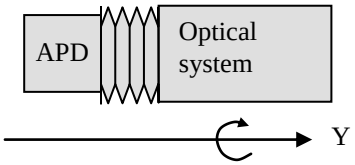


2. APD array has 3 DOF like the picture below.
   The APD array can be adjusted manually in three degree of freedom.
   The movable range of the APD array is:
   - X: 3mm
   - Z: 5mm
   - Rotation: 5˚

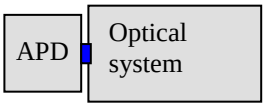


It should have an optically sealed attachment for the APD that allows for a few mm of translation and rotation. A few possibilities for solving this problem are given below

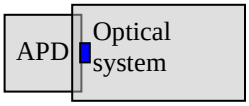
Option 1:
This option utilizes a soft collapsible membrane that covers the junction between the APD array and the optical system. By using a soft collapsible membrane, the APD array can be adjusted with 3 degrees of freedom (X, Z, and rotation about the Y) without deflection of the connection and loss of the seal.
The rear rectangular opening in the optical system is 18.5mm by 33mm with accordance to the ADP Array dimensions. Please look at the APD Array specification for more details.



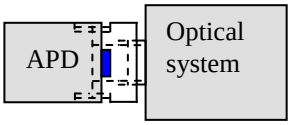
Option 2:
This option utilizes rubber to create a seal around the juction of the APD array and the optical system.



Option3:
The APD array can be placed inside of the optical system casing. The main problems posing this option is the ability to adjust the APD array while it is within the optical casing.



Option4:
A C-mount optical component is used for the juction between APD array and optical system.

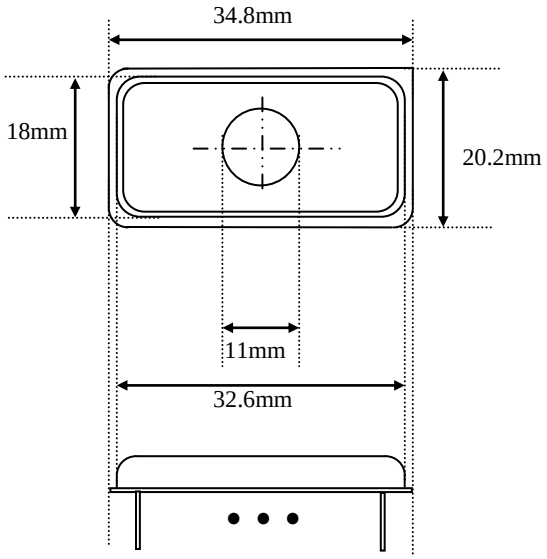


APD specification

Mechanical Characteristics:

Elements: 25
Total active length: 7.5mm
Active width: 0.3mm
Center-to-center spacing: 0.3mm
Dead space between elements: 75μm

Dimensions:



Optical Characteristics:

Center Wavelength: 905±10nm
Time of Response
Rise time typically(R_L=50Ω, λ=905nm) 2ns
Fall time typically(R_L=50Ω, λ=905nm) 2ns
Quantum efficiency:
85%typically at 905nm
18%typically at 1060nm
Spectral response Rang(10% Points)
400 to 1100nm
Operating Temperature Range
-40˚C to +80˚C

Electrical Characteristics:

Operating Voltage: -275 to -425V, typically (–300V)
Reverse Bias Current, Total: 200μA(Max)
Element-to-Element
Gain Non-uniformity: ±5%
Dark Current:
Guard Ring: 100nA, 300nA (Max)
Each element: 1nA
Responsivity:

At 900nm:	(min) 25A/W, 31(A/W)
At 1060nm:	7.5 (A/W)
Capacitance:	
Total:	25 pf
Each element:	0.5 pf
Interelectrode	0.2 pf
Series Resistance:	
Each element:	100Ω
Noise Current, I_n:	
f=10kHz,Δf=1.0Hz	
All elements	0.5pA/Hz$^{1/2}$
Each element	0.1pA/Hz$^{1/2}$

Other Characteristics:

Refer to the specification of the EG&G C30985E avalanche photodiode array.

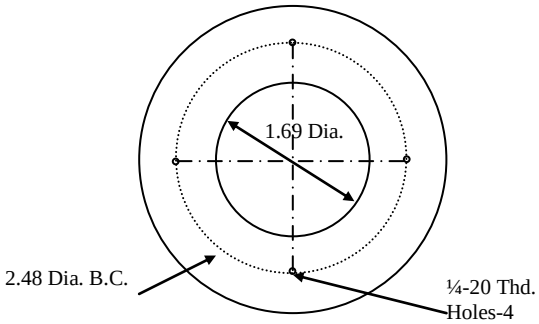
End product specifications

Dimensions:

Units: inches

The end product should be mounted on an optical holder with the 4 holes as shown in the figure.

The dimension of the product must be less than 2.5 x 2.9 x 2.5 inches cubed



Optical holder

The APD array is mounted in the center of the Optical holder

Output Characteristics of the Pre-amplifier:

Common-mode rejection ratio	(typically) 80dB
Supply voltage rejection ratio	
($\Delta V_{cc}/\Delta V_{IO}$)	(typically) 70dB
Noise Figure:	<1.4dB

Output of End Product *

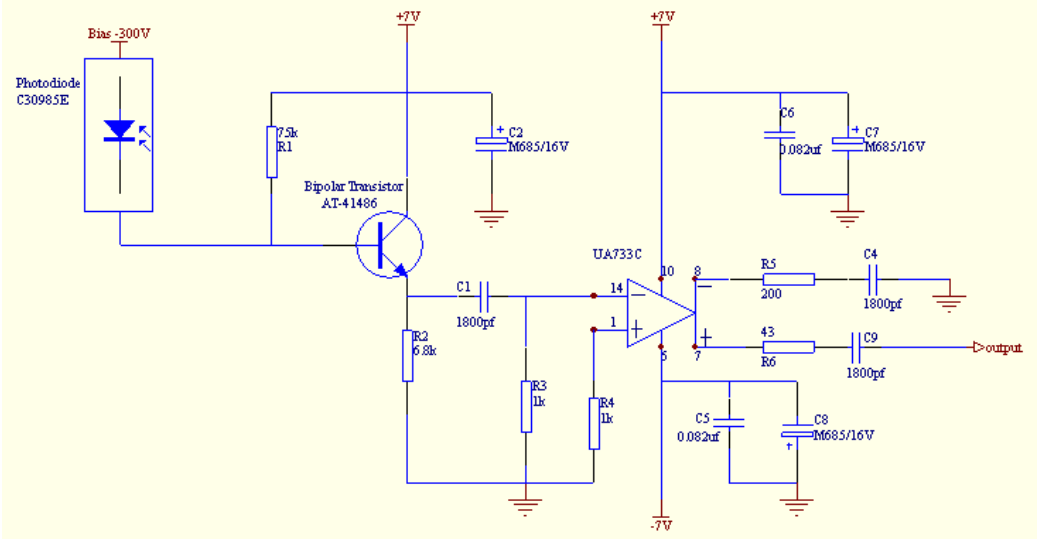
Output (at 10Khz pulse laser)	
Voltage (peak)	>1.5v
Output Current	10mA
(peak)	>20mA
Sink Current	3.6mA
Output Resistance:	<50Ω
SNR_{out}:	>30dB

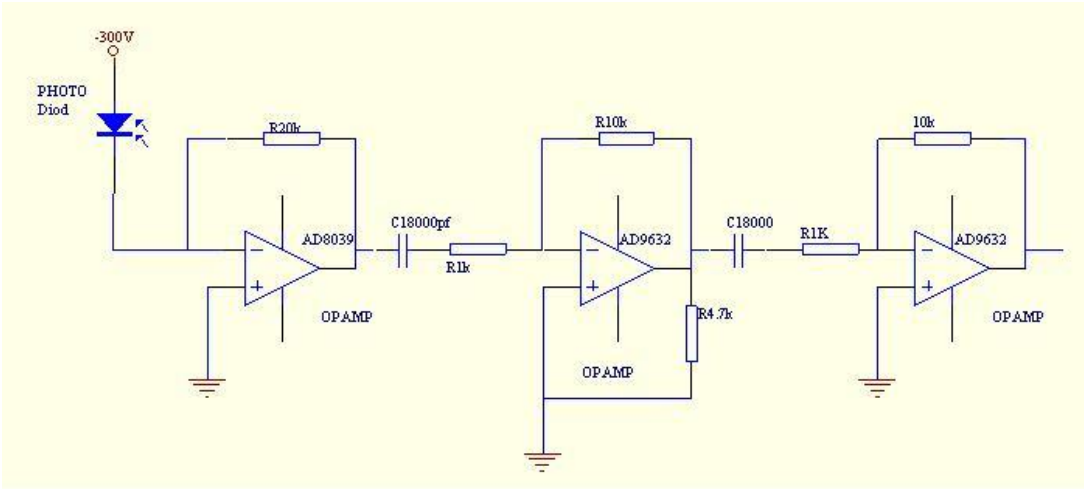
* Signal condition is:
On Highway.

Length of laser path (reflected from the ground surface) is 20 meters
Laser characteristics:

pulse repetition frequency	10Khz
pulse width fixed at	9ns
output power(peak)	20W
wavelength of laser	905nm

Two sample pre-amplifier circuits are shown below.





The top schematic is the currently used schematic. The bottom schematic is not utilized due to high power consumption. The circuitry heats up too much and causes a lot of heating noise. The bottom schematic is also expensive to produce compared to the top one.

Appendix C: Eye-Safety Calculations

Laser eye safety calculations for peak power 60W Laser with 10 kHz and 905 nm wavelength:

According to ANSI Z136.1-2000 8.2.3 Repeated Exposures, there are 3 rules for the Maximum Permissible Exposure (MPE).

Rule 1: *Single-Pulse MPE Limit*:

$MPE_{sp} = 0.5C_A*10^{-6}$ J.cm^{-2}

Where $C_A=10^{2(\lambda-0.700)}=2.57$ ($\lambda=0.905$)
$= 0.5 * 2.57 *10^{-6}$ J.cm^{-2}
$= 1.29 * 10^{-6}$ J.cm^{-2}

Rule 2: *Average Power Limit*:

$MPE:H_{group} = 1.8 * 10^{-3} C_A t^{0.75}$ J.cm^{-2}

Where t = 10 s
$=1.8 * 10^{-3} * 2.57 * 10^{-0.75}$
$=2.6 * 10^{-2}$ J.cm^{-2}
$MPE/pulse = MPE:H_{group}/10*10k$
$= 2.6 * 10^{-7}$ J.cm^{-2}

Rule 3: *Repetitive Pulse Limit*:

$MPE/pulse = n^{-0.25} MPE_{sp}$

Where $n = 10(s)*10K=10^5$
$=10^{5*0.25} * 0.5* 2.57 *10^{-6}$
$=7.23 * 10^{-8}$ J.cm^{-2}

The resultant MPE per pulse should be the minimum of the calculated values shown above.

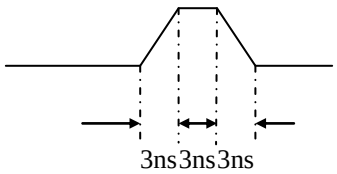
Therefore,

$MPE = 7.23 * 10^{-8}$ J cm^{-2}.

So the average irradiance is,

$MPE:E = 10$ (s) $* 7.23*10^{-8}$ (J cm^{-2}) $*$ 10 (kHz) / 10 (s)
$= 7.23*10^{-4}$ (W cm^{-2}).

Given a laser with 60W (peak) and 9ns duration like below:



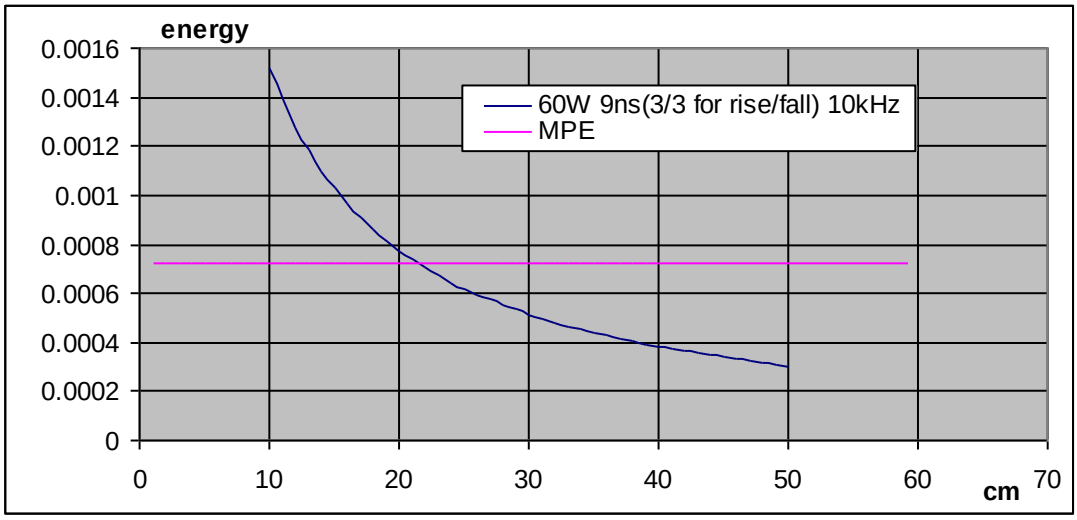
The pulse energy is $\mathbf{P_p}$:

$\mathbf{P_p} = (3/2+3+3/2)*10^{-9}*60$ W $=3.6 * 10^{-7}$ J.pulse^{-1}

The average energy of the laser power **P** :

$$\mathbf{P} = \mathbf{P_p} * 10 * 10^3 \text{ Hz}$$
$$= 3.6 * 10^{-3} \text{ W}$$

Figure 1 shows the relationship between the energy per area of the laser and the distance away from the front of the laser.



As can be seen from the figure, the eye-safe distance starts at 22 cm. This is acceptable due to the fact that the lasers will be located 22 cm into the device as seen from the front. The length of laser line at the front of frame is about 25.977cm (Distance = 35cm) shown in Table 2, below.

D	S	L	W	P	MPE
20	4.649056	15.05881		0.000774	0.000725
21	4.880679	15.78675		0.000738	0.000725
22	5.112937	16.51469		0.000704	0.000725
23	5.34583	17.24263		0.000673	0.000725
24	5.579359	17.97057		0.000645	0.000725
25	5.813523	18.69851		0.000619	0.000725
26	6.048322	19.42645		0.000595	0.000725
27	6.283756	20.15439		0.000573	0.000725
28	6.519826	20.88233		0.000552	0.000725
29	6.756531	21.61027		0.000533	0.000725
30	6.993871	22.33821		0.000515	0.000725
31	7.231846	23.06615		0.000498	0.000725
32	7.470457	23.79409		0.000482	0.000725
33	7.709702	24.52204		0.000467	0.000725
34	7.949584	25.24998		0.000453	0.000725
35	8.1901	25.97792		0.00044	0.000725

Appendix D: Source Code

Source Code in microchip for laser pulse and timing window

defreg.h

```
/************************************************
 * define variable and special function register
 * only for pic16F876 according to p16F876.inc
 *
 *  by  :  zhaoqing Wang
 *  date:  11/08/2001
 ************************************************/

// bank0 registers
// char INDF      @0x0000; //already defined
// char TMR0      @0x0001; //already defined
// char PCL @0x0002; //already defined
// char STATUS    @0x0003; //already defined
// char FSR @0x0004; //already defined
// char PORTA     @x0005;  //already defined
// char PORTB     @0x0006; //already defined

char PORTC  @0x0007;

// char PCLATH     @0x000A; //already defined
// char INTCON     @0x000B; //already defined
int  TMR1   @0x000E;
char PIR1   @0x000C;
char PIR2   @0x000D;
char TMR1L  @0x000E;
char TMR1H  @0x000F;
char T1CON  @0x0010;
char TMR2   @0x0011;
char T2CON  @0x0012;
char SSPBUF @0x0013;
char SSPCON @0x0014;
char CCPR1L @0x0015;
char CCPR1H @0x0016;
char CCP1CON      @0x0017;
char RCSTA  @0x0018;
char TXREG  @0x0019;
char RCREG  @0x001A;
char CCPR2L @0x001B;
char CCPR2H @0x001C;
char CCP2CON      @0x001D;
char ADRESH @0x001E;
char ADCON0 @0x001F;

// bank1 registers
```

lasr.h

```
void Init();
void Pwm();
```

laserdef.h

```
// for init port a,b,c

#define  TRIS_B 0x80     //<7>pins of port b as interrupt
input,<0:6> pins as output
#define  TRIS_C 0x01     // pin <0> as input, other as output
#define  TRIS_A 0x00     // all pins<0:7> of port a as input


// copy from tt1.asm written by hang

#define OPTION_P  0x4e  //Presale assign to WDT with 1:64
#define INT_INT_P 0X90  //Enable int/RB0  int
#define RB47_INT_P      0x88  //Enable DB4-7 int
#define PIE1_P          0X00  //Disable all peripheral int
#define PIE2_P          0X00  //Disable other int
#define PCON_P          0x03  //clr POR/BOR
#define ADCON1_P  0x06  //Set porta as digital I/O

#define PERIOD_FIL1     0X03  //PM tuning filter, contiuously 3
times period
                        //correct, then PM is OK
#define PERIOD_FIL2     0X10  //Period detect filter
#define T1CON_P         0x31    //Prescale 1:8, enable timer1.P51
/*******************************************************
 * the value of T1CON_INIT is calculated as below:
 *      Enable tmr1 with internal clk, prescale=8
 *    max(TMR1L)=256*Tosc*4*8=512us  Tosc=1/20MHz=50ns
 *******************************************************/
#define T2CON_P         0x06    //Enable TMR2, Prescale=16,
postscale=0. P55

/*=================================================================
=======
 *PWM PARAMETER
 *PWM Period=[(pr2)+1]*4*Tosc*(TMR2 prescale value=16)
 *Tosc=50ns, pr2=1F then PWM period=10kHz. P61
 *================================================================*/
#define PR2_P           0x1f

/*=================================================================
=======
PWM duty circle=(CCPR1L:CCP1CON<5:4>).Tosc.(TMR2 prescale valve=16)
  Tosc=50ns, TMR2 Precscal valve=4,
(CCPR1L:CCP1CON<5:4>)=13=0dh=000011.01
  ie: CCPR1L=0X06 & CCP1CON<5:4>=0X01 then PWM duty cir=10us. P61
================================================================*/

/* define the duty circle as 50us */

#define CCPR1L_P  0x0f

/* bit3~0 :11xx= PWM mode. P58 */

#define CCP1CON_P       0x2C

//#define RBIF          0       //RB0 interrupt
```

```
#define DIFF_P          0x03    //the torelance of period
#define PERIOD_P  78     //(0x26+1)*2

#define PERIOD_U  81     //PERIOD_P+DIFF_P
#define PERIOD_D  75     //PERIOD_P-DIFF_P

//#define FILT_TIMEOUT    780    //PERIOD_P*10

#define TIMEDUTY  156   //~PERIOD_P*2   timeout of interrupt

/*================================================================
   output(interrupt) period
   =[(pr2)+1]*4*Tosc*(TMR2 prescale value=16)/
          ((TMR1 prescale value=8)*4*Tosc(TMR))
   =78, timeout =. 2*period, so TIMEOUT take 150.
================================================================*/

#define BIT0            0      //set bit
#define BIT1            1
#define BIT2            2
#define BIT3            3
#define BIT4            4
#define BIT5            5
#define BIT6            6
#define BIT7            7

#define TRUE            1
#define FALSE           0
#define LOW_LEVEL 0
#define HIGH_LEVEL      1

/*****************************************
 *  EOF
 *****************************************/
```

pulsepower.c

```
/*****************************************************************
********/
/*
*/
/* This program is for Microchip PIC16F876 to generate 4 pulses and
*/
/* 2 time windows. Outputs of B6 B5 are for one side laser supply.
*/
/* B4 for their time windows. Outputs of B2 and B1 are for another
side    */
/* laser pulse supply. B3 is their time windows
*/
/*                                                          */
/* By Zhaoqing Wang                                             */
/* Date:09/04/2002                                              */
/* version :1.0                                                 */
/*                                                          */
/* modified By qingcang Yu                                      */
/* Date:09/27/2002                                              */
```

```
/* version :1.0
*/
/*
*/
/* updated by zhaoqing wang
*/
/* date:08/20/2003
*/
/* when the Pin C0 is 0, it means a push button is push down. the
laser      */
/* source should be closed for microchip to measure the noise
without signal */
/* after 5 seconds, it turn on the laser again.
*/
/*
*/
/*****************************************************************
**********/


#include "laser.h"     // Function prototype declearations
#include "laserdef.h"  // Definitions of constants and initial
values of
                       // special function registers
#include "defreg.h"    // Defination of addresses of
                       // all available special function registers

/*********************************¡*********************************
*/
/* interrupt handler: interrupt()
*/
/*
*/
/* By : Zhaoqing Wang
*/
/* date: 08/14/2001
*/
/*
*/
/*
*/
/* The handler stores the 16-bit time stamp and portb status
*/
/* in an array of 24 bytes, when a interrupt from PortB <4:7> is
*/
/* received. The time stamp is stored in the first two byte and
*/
/* the status is stored in the third byte. Therefore the array can
*/
/* store time stamps and status for 8 interrupts.
*/
/*
*/
/*
*/
/*****************************************************************
*/
```

```
char flag = 0;
char timewindow_flg = 1;

int  read_tmr1;         //return from ReadTime().
char read_tmr1l @0x40;  //low byte of 16-bit time.
char read_tmr1h @0x41;  //high byte of 16-bit time.
char pinc;

void interrupt( void )
{
    if (flag == 0)
    {
        if( pinc == LOW_LEVEL ){
          output_low_port_b(BIT2);nop();nop();nop();nop();nop();
            nop();
            output_port_b( 0x3B);
            nop();nop();nop();nop();
          output_high_port_b(BIT2);
        }
        else{
          output_low_port_b(BIT2);nop();nop();nop();nop();nop();
            output_port_b( 0x38 );
            nop();nop();nop();nop();
          output_high_port_b(BIT2);
        }
        /* set time window */
        flag = 1;
   }
   else
   {
        if( pinc == LOW_LEVEL){
          output_low_port_b(BIT3);nop();nop();nop();nop();nop();
            nop();
            output_port_b(0x37);
            nop();nop();nop();nop();
          output_high_port_b(BIT3);
        }
        else{
          output_low_port_b(BIT3);nop();nop();nop();nop();nop();
            output_port_b( 0x07 );
            nop();nop();nop();nop();
          output_high_port_b(BIT3);
        }
        /* set time window */
        flag = 0;
    }
    clear_bit( INTCON, RBIF ); // clear portb<7> interrupt flag

}

void Pwm()
{
    /* refer to manual P62 */
    /* 1. set PWM period by writing to the PR2 register */
    PR2 = PR2_P;
    /* 2. set the PWM duty cycle by writing to CCPR1L register
```

```
          and CCP1CON<5:4> bits
    */
    CCPR1L = CCPR1L_P;

    clear_bit( CCP1CON, BIT4);
    set_bit( CCP1CON, BIT5);

    /* Make the CCP1 pin as output by clearing the TRISC <2> bit */
    clear_bit( TRISC, BIT2);

    /* set the TMR2 prescal value and enable Timer2 by writing to
T2CON */
    T2CON = T2CON_P;

    /* Configure the CCP1 module for PWM operation */
    CCP1CON = CCP1CON_P;
}

/*********************************************************/
/* Init()                                                */
/*                                                       */
/* This function initializes the Microchip 16f876 system   */
/*                                                       */
/* By :   zhaoqing wang                                  */
/* date :   09/04/2002                                   */
/*                                                       */
/*********************************************************/

void Init()
{

    ADCON1 = ADCON1_P;
    OPTION_REG = OPTION_P;
    PIE1 = PIE1_P;
    PIE2 = PIE2_P;
    PCON = PCON_P;
    T1CON = T1CON_P;
    set_tris_b( TRIS_B );       // set port B <7> as input of
interrrupt.
                                // set port B <0:6> as output.
    set_tris_c( TRIS_C );
    output_port_b( 0x03 );

    enable_interrupt( RBIE );   // enable RB port change interrupt
bit
    disable_interrupt( GIE );   // enable global interrupt bit

    Pwm();
    enable_interrupt( GIE);

}


/*************************************************/
/* the main program                              */
/*                                               */
/*    By : zhaoqing Wang                         */
```

```
/*   date: 09/04/2002                                      */
/*   waiting interrup and output portb    circlely   */
/*                                                        */
/**********************************************************/

main()
{
    int time_window_duty;

    delay_s(5);         // if system comes to steady, it start to
work.
    Init();

    enable_interrupt(RBIE);
    enable_interrupt(GIE);
    while ( TRUE )
    {
       pinc = input_pin_port_c(BIT0);
       if( pinc==LOW_LEVEL ){
         delay_s(15);
       }
    }
}

/*****************
    EOF
*****************/
```

Source Code in microchip

DEFREG.H

```
/*****************************************************
 * define variable and special function register
 * only for pic16F876 according to p16F876.inc
 *
 *  by : zhaoqing Wang
 *  date: 11/08/2001
 *****************************************************/

// bank0 registers
// char INDF      @0x0000; //already defined
// char TMR0      @0x0001; //already defined
// char PCL       @0x0002; //already defined
// char STATUS    @0x0003; //already defined
// char FSR       @0x0004; //already defined
// char PORTA     @x0005; //already defined
// char PORTB     @0x0006; //already defined

char PORTC       @0x0007;

// char PCLATH     @0x000A; //already defined
// char INTCON     @0x000B; //already defined
int  TMR1        @0x000E;
char PIR1        @0x000C;
char PIR2        @0x000D;
```

```
char TMR1L       @0x000E;
char TMR1H       @0x000F;
char T1CON       @0x0010;
char TMR2        @0x0011;
char T2CON       @0x0012;
char SSPBUF      @0x0013;
char SSPCON      @0x0014;
char CCPR1L      @0x0015;
char CCPR1H      @0x0016;
char CCP1CON     @0x0017;
char RCSTA       @0x0018;
char TXREG       @0x0019;
char RCREG       @0x001A;
char CCPR2L      @0x001B;
char CCPR2H      @0x001C;
char CCP2CON     @0x001D;
char ADRESH      @0x001E;
char ADCON0      @0x001F;

// bank1 registers

//char OPTION_REG @0x0081; //already defined
//char TRISA      @0x0085; //already defined
//char TRISB      @0x0086; //already defined
char TRISC      @0x0087; //already defined
char PIE1       @0x008C;
char PIE2       @0x008D;
char PCON       @0x008E;
char SSPCON2    @0x0091;
char PR2        @0x0092;
char SSPADD     @0x0093;
char SSPSTAT    @0x0094;
char TXSTA      @0x0098;
char SPBRG      @0x0099;
char ADRESL     @0x009E;
char ADCON1     @0x009F;

//bank2 register
//char EEDATA     @0x010C; //already defined
//char EEADR      @0x010D; //already defined
char EEDATH     @0x010E;
char EEADRH     @0x010F;

//bank3 register
//char EECON1     @0x018C; //already defined
//char EECON2     @0x018D; //already defined

/** End of File **/
```

LASER.H

```
/*****************************************************************
 *  Function Prototypes
 *****************************************************************/
```

```
void XdcpGoZero();
void XdcpGoMax();
void XdcpSet();
void XdcpGo1Step();
void XdcpGoHalf();
void DetectPeriod();
void Init();
void Pwm();
void XdcpIntHandling();
void FlashLed();
void LightLed();
void DarkLed();
void FiltIntHandling();
void ReadTime();
void C2Int();
void Filter();
void WriteByteToEE();
void ReadByteFromEE();
void XdcpGo0();
/** End of File **/
```

LASERDEF.H

```
/************************************
 *  Register Definitions for Ports
 ************************************/
// for init port a,b,c

#define  TRIS_B 0xF2    //<1,4:7>pins of port b as interrupt input,<0,2,3> pins as output
#define  TRIS_C 0x01    // pin <0> as input, other as output
#define  TRIS_A 0x00    // all pins<0:7> of port a as input


// copy from tt1.asm wroted by zhang
#define OPTION_P        0x4e    //Prescale assign to WDT with 1:64
#define INT_INT_P       0X90    //Enable int/RB0  int
#define RB47_INT_P      0x88    //Enable DB4-7 int
#define PIE1_P          0X00    //Disable all peripheral int
```

```
#define PIE2_P              0X00     //Disable other int
#define PCON_P              0x03     //clr POR/BOR
#define ADCON1_P            0x06     //Set porta as digital I/O
#define PERIOD_FIL1         0X03     //PM tunning filter,contiuously 3 times period
                                     //correct, then PM is OK
#define PERIOD_FIL2         0X10     //Period detect filter
#define T1CON_P         0x31    //Prescale 1:8, enable timer1.P51
/*********************************************************
 * the value of T1CON_INIT is calculated as below:
 * Enable tmr1 with internal clk, prescale=8
 * max(TMR1L)=256*Tosc*4*8=512us  Tosc=1/16MHz
 *********************************************************/
#define T2CON_P         0x06    //Enable TMR2, Prescale=16, postscale=0. P55
/*=====================================================================
 *PWM PARAMETER
 *PWM Period=[(pr2)+1]*4*Tosc*(TMR2 prescale value=16)
 *Tosc=100ns, pr2=26 then PWM period=5000Hz
 *=====================================================================*/
#define PR2_P               0x26     //0x26

/*=====================================================================
PWM duty circle=(CCPR1L:CCP1CON<5:4>).Tosc.(TMR2 prescale valve=16)
  Tosc=100ns, TMR2 Precscal valve=4, (CCPR1L:CCP1CON<5:4>)=13=0dh=000011.01
  ie: CCPR1L=0X06 & CCP1CON<5:4>=0X01 then PWM duty cir=10us
=====================================================================*/

#define CCPR1L_P            0x03
#define CCP1CON_P           0x1C     //Set PWM Mode and clr CCPxX:CCPxY
//#define RBIF              0       //RB0 interrupt

#define DIFF_P              0x03    //the torelance of period
#define PERIOD_P            78     //(0x26+1)*2

#define PERIOD_U            81     //PERIOD_P+DIFF_P
#define PERIOD_D            75     //PERIOD_P-DIFF_P

#define FILT_TIMEOUT    780    //PERIOD_P*10
```

```
//#define FILT_TIMEOUT    468      //PERIOD_P*6
#define FILT_NUM          21

#define TIMEOUT                   156         //~PERIOD_P*2   timeout of interrupt

/*================================================================
  output(interrupt) period
  =[(pr2)+1]*4*Tosc*(TMR2 prescale value=16)/
       ((TMR1 prescale value=8)*4*Tosc(TMR))
  =78, timeout =. 2*period, so TIMEOUT take 150.
================================================================*/


#define XDCP_FINISHED_NUM 3     //continuous times of adjusting the xdcp to ensure the output
in
                        //required period.

#define BIT0              0     //set bit
#define BIT1              1
#define BIT2              2
#define BIT3              3
#define BIT4              4
#define BIT5              5
#define BIT6              6
#define BIT7              7

#define BIT_IMG0          0x01
#define BIT_IMG1          0x02
#define BIT_IMG2          0x04
#define BIT_IMG3          0x08
#define BIT_IMG4          0x10
#define BIT_IMG5          0x20
#define BIT_IMG6          0x40
#define BIT_IMG7          0x80

#define LOW_LEVEL     0
#define HIGH_LEVEL    1
```

```
#define TRUE            1
#define FALSE           0
#define XDCP_INIT       0
#define XDCP_WORK       1
#define XDCP_ERR        2
#define SPEED_NO        50
#define SPEED_SOME      100
#define SPEED_ALL       200
#define MAX_STEP        100    //total steps of XDCP
#define XDCP_ADDR       0x00
#define SPEED_ADDR      0x01

#define INT_MAX_NUM 23     //the max number of interrepter
#define MIN_XDCPSET  10000    //the min time for continue high level of all channels

#define NO_BT           0
#define PINB1           1
#define PINC0           2

#define MIN_STEPS       0     //minimum steps for go down each channel
#define UNSTABLE_STEPS  1    //too make sure output is stable when no laser signal
/** End of File **/
```

NEWLASER.C

```
/*****************************************************************/
/*
*/
/* This program is for Microchip PIC16F876 to control X9312
*/
/* Digitally Controlled Potentiometer (XDCP)and data pre-processing
*/
/* of signals from sensor electronics.
*/
/*
*/
/* Functionality:
*/
/*   1) Set the XDCPs values to achieve proper signal gain of signal
*/
/*      detection circuit.
*/
/*   2) Filter out the spikes and noise from the detcted signals.
*/
```

```
/*
*/
/*
*/
/* By : Zhaoqing Wang
*/
/* Date : 08/11/2001
*/
/* Version : 2.0
*/
/*
*/
/* The updated version 2.0. differs from the V1.0 in the algorism of
*/
/* signal processing. In version 1.0, the period is used to
distiguish     */
/* noise from signal. In the version 2.0, the noise is detected by
*/
/* measuring the duration of signal. If the duration of signal is
less     */
/* than a certain amount of time tick, it is considered as noise and
*/
/* will be filtered out.
*/
/*
*/
/* Update date : 08/31/2001
*/
/*
*/
/*
*/
/* A LED which is connected to RC7 is used to indicated the
adjustment     */
/* status of XDCP. If system works normally, it will be on. If there
is    */
/* an error in setting XDCP, it will be blinking. The frequency of
blinking*/
/* depends on the values of XDCP when error occurs.
*/
/*
*/
/* The output of sensors electronics is feeded back to PIC16F876
*/
/* portb <4:7>. The processed signals are output to the host
computer      */
/* from portb <0:3>. The enable signal (cs) is connected to port A
<0:3>   */
/*
*/
/*-------------------------------------------------------------------
----------*/
/* updated by zhaoqing wang, 07/03/2003
*/
/* portc <4:7> are as output to the rabbit processors or computer
*/
```

```
/* portb <3> is as the director of the LED
*/
/* the potentiometer will be set by the synchronization signal (time
windows) */
/*
*/
/*****************************************************************
********/

#include "laser.h"     // Function prototype declearations
#include "laserdef.h"  // Definitions of constants and initial
values of
                       // special function registers
#include "defreg.h"    // Defination of addresses of
                       // all available special function registers

char i;
int j;
int k;
char flash_speed;       // Frequency of LED blinking
char temp_port;

char interrupt_time_status[24]; // Array to store time and port
status for 8
                                // interrupts, each interrupt takes
3 bytes.

char int_mem_offset;    //offset of interrupt_time_status array.
char interrupt_handle;  //number of interrupts.

int  read_tmr1;         //return from ReadTime().
char read_tmr1l @0x40;  //low byte of 16-bit time.
char read_tmr1h @0x41;  //high byte of 16-bit time.

char xdcp_cs_buffer;    //select channel of the xdcps.
                        //If the bit is 0, the channel is selected.
char xdcp_ok;           //for xdcp adjusting status

char filt_status;       //Port B status. If the status changed,
                        //the correspoding bit set to 0.

int old_tmr11;          //the time of last interrupt in channel 1
int old_tmr12;
int old_tmr13;
int old_tmr14;

int int_temp;
int temp_int_time;      //Used in function C2Int()

char xdcp_counter;      //counter of going up steps of XDCP.

char old_status;        //last interrupt status in Filter()
char diff_status;       //Exclusive Or between current and old
filter status.

char eedata;
char eeaddress;
```

```
char c0;
char c1;
char c2;
char c3;
char bc0;
char bc1;
char bc2;
char bc3;
int cmin;
char button;
char flg_int;
/*********************************¡*********************************
*/
/* interrupt handler: interrupt()
*/
/*
*/
/* By : Zhaoqing Wang
*/
/* date: 08/14/2001
*/
/*
*/
/*
*/
/* The handler stores the 16-bit time stamp and portb status
*/
/* in an array of 24 bytes, when a interrupt from PortB <4:7> is
*/
/* received. The time stamp is stored in the first two byte and
*/
/* the status is stored in the third byte. Therefore the array can
*/
/* store time stamps and status for 8 interrupts.
*/
/*
*/
/*
*/
/*
*/
/*********************************************************************
*/

void interrupt( void )
{
    ReadTime();

    interrupt_time_status[int_mem_offset] = read_tmr1l;
    int_mem_offset++;
    interrupt_time_status[int_mem_offset] = read_tmr1h;

    int_mem_offset++;
    interrupt_time_status[int_mem_offset] = input_port_b();

    interrupt_handle++;
    int_mem_offset++;
    if (int_mem_offset >=INT_MAX_NUM)
        int_mem_offset = 0;
```

```
    //int_mem_offset &= 0x07;  //0x0f;   int_mem_offset mod 16 ;for
old version
    clear_bit( INTCON, RBIF ); // clear portb<4:7> interrupt flag

}

/*************************************************************/
/* XdcpSet()                                                 */
/*                                                           */
/* XDCP will be set to proper value so that laser signal can */
/* be detected by digital output circuit.                    */
/*                                                           */
/*                                                           */
/* by   : zhaoqing Wang                                      */
/* date : 08/14/2001                                         */
/* update: 08/27/2001  07/21/2003                            */
/* 07/21/2003:                                               */
/*   At first, it sets the XDCP to 0. it means at this point,*/
/*      the synchronal signal will the max to be compared.   */
/*      The output from the microchip should be HIGH, otherw-*/
/*      ise maybe there is no signal coming.                 */
/*   Then, it will increace the potentiometer 100ohm per step*/
/*      till the output from the microchip is low. While inc-*/
/*      reasing the reisister, the steps is record in the c* */
/*      variables. cmin is the minimum steps of all channels */
/*   At last, it adjusts the potentiometer to the proper     */
/*      position at which the output from the microchip is   */
/*      HIGH with the almost maximum possible potentiometer  */
/*      value. The SCALE is defined as the coefficient of    */
/*      c*                                                   */
/*                                                           */
/*************************************************************/

void XdcpSet()
{

    disable_interrupt(GIE);

    i = 0;
    int_mem_offset = 0;
    interrupt_handle = 0;
    xdcp_counter = 0;
    c0=0;
    c1=0;
    c2=0;
    c3=0;
    cmin=0;
    flg_int=0x0f;

    XdcpGoZero();           // XDCP is set to 0 value.
    delay_s(2);             // Wait for the system to become stable.

    xdcp_cs_buffer = 0x30;  // all channels are selected.
    old_status = input_port_b();
    output_port_c(old_status);
    clear_bit(INTCON, RBIF);
```

```
    enable_interrupt(GIE);

    if ((old_status & 0xf0) == 0x00 )
    {
        xdcp_ok = XDCP_ERR;
        flash_speed = SPEED_NO;
        return;
    }
    if ((old_status & 0xf0) != 0xf0 )
    {
        xdcp_ok = XDCP_ERR;
        flash_speed = SPEED_SOME;
    }

    while( TRUE )
    {
        if( interrupt_handle>0 )      // At least one channel is
ready.
        {

           XdcpIntHandling();
           interrupt_handle--;
        }

        if ( (xdcp_cs_buffer & 0x0f)==0x0f ) // all potentiometers
are ready.
        {
           if( (cmin < UNSTABLE_STEPS) && (button == PINB1)){
              cmin++;
              continue;
           }
           else{
             xdcp_ok = XDCP_WORK;
             break;
           }
        }
      else{
           cmin=0;
        }
        output_port_a( xdcp_cs_buffer); // increase 100ohm
        XdcpGo1Step();
        //count the increasing steps of each channel
        if( ~xdcp_cs_buffer & BIT_IMG0 )
            c0++;
        if( ~xdcp_cs_buffer & BIT_IMG1 )
            c1++;
        if( ~xdcp_cs_buffer & BIT_IMG2 )
            c2++;
        if( ~xdcp_cs_buffer & BIT_IMG3 )
            c3++;

        delay_ms(100);          //delay some times for store the
position
        output_port_a(0x30);
        XdcpGo1Step();
        c0++;
        c1++;
```

```
        c2++;
        c3++;
        delay_ms(100);
        output_port_a( 0x3f);   //store wiper position
        delay_ms(50);

        xdcp_counter++;
        if (xdcp_counter >= MAX_STEP) // xdcp_counter>100 steps
indicates an error.
        {
           xdcp_ok = XDCP_ERR;        // set the LED as error
status.
           output_port_a( 0x3f );     // store wiper position

           if (xdcp_cs_buffer == 0x30){
               flash_speed = SPEED_ALL;
           }
           else{
               flash_speed = SPEED_SOME;
           }
           break;
        }
    }

    disable_interrupt(GIE);
    if( button == PINB1 ){
        bc0=c0;
        bc1=c1;
        bc2=c2;
        bc3=c3;
    }

    if( button == PINC0 ){

        if(c0>bc0){
           c0 = c0-bc0;
           if(c0>50)
               c0=c0-15;
           else{
               c0 = c0/4;
               c0 = c0*3;
           }
        }
        else
           c0=0;

        if(c1>bc1){
           c1 = c1-bc1;
           if(c1>50)
               c1=c1-15;
           else{
               c1 = c1/4;
               c1 = c1*3;

           }
        }
        else
```

```
   c1=0;

if(c2>bc2){
   c2 = c2-bc2;
   if(c2>50)
       c2=c2-15;
   else{
       c2 = c2/4;
       c2 = c2*3;
   }
}
else
   c2=0;

if(c3>bc3){
   c3 = c3-bc3;
   if(c3>50)
       c3=c3-15;
   else{
       c3 = c3/2;
       c3 = c3*3;
       c3 = c3/2;
   }
}
else
   c3=0;

if(c0 > MIN_STEPS ){
  output_port_a(0x2E);  //PA0 decreace
  for( k=0; k < c0; k++)
    XdcpGo1Step();
}
else{
  output_port_a(0x2E);
  XdcpGo0();
}
output_port_a(0x3f);  //PA3 decreace
if(c1 > MIN_STEPS ){
  output_port_a(0x2D);  //PA1 decreace
  for( k=0; k < c1; k++)
    XdcpGo1Step();
}
else{
  output_port_a(0x2D);
  XdcpGo0();
}
output_port_a(0x3f);  //PA3 decreace
if(c2 > MIN_STEPS ){
  output_port_a(0x2B);  //PA2 decreace
  for( k=0; k < c2; k++)
    XdcpGo1Step();
}
else{
  output_port_a(0x2B);
  XdcpGo0();
}
output_port_a(0x3f);  //PA3 decreace
```

```
        if(c3 > MIN_STEPS ){
          output_port_a(0x27);  //PA3 decreace
          for( k=0; k < c3; k++)
            XdcpGo1Step();
        }
        else{
          output_port_a(0x27);
          XdcpGo0();
        }
    }
    clear_bit(INTCON, RBIF);
    delay_ms(3);
    output_port_a(0x3f);  //PA3 decreace
    delay_s(1);
    enable_interrupt(GIE);
}

/*****************************************************************
/
/* XdcpIntHandling()
*/
/*
*/
/* When the adjust button is pushed, it generates an interrupt
and.*/
/* the interrupt counter is incremented. This function is called
*/
/* by XdcpSet() when the interrput counter is not 0.
*/
/*
*/
/* This function reads data from interrupt status array and
*/
/* dtermine which channel(s) cause the interrupt and set the
*/
/* channel selection varible xdcp_cs_buffer.
*/
/*
*/
/* by   :  zhaoqing wang
*/
/* date :  08/14/2001
*/
/* update :08/27/2001
*/
/*
*/
/*
*/
/*****************************************************************
/

void XdcpIntHandling()
{
     char temp_int_status;

     i++;
```

```
     i++;       // first 2 char is for time stamp, the 3rd is for
portb status
     temp_int_status = interrupt_time_status[i];
     diff_status = old_status ^ temp_int_status;
     i++;
     if( i>=INT_MAX_NUM )
         i = 0;

     // for channel 1 changed
     if (diff_status & BIT_IMG4){
       if( old_status & BIT_IMG4 )        // channel 1 changed from
HIGH to LOW(down)
          set_bit(xdcp_cs_buffer,BIT0);   // set the bit of channel
selection varible.
/*       if(temp_int_status & BIT_IMG4)     // channel 1 changed
(up)
          if(button == PINB1)
            clear_bit(xdcp_cs_buffer,BIT0);
*/
     }
     // for channel 2
     if (diff_status & BIT_IMG5 ){
       if( old_status & BIT_IMG5 )
         set_bit(xdcp_cs_buffer,BIT1);
/*       if(temp_int_status & BIT_IMG5)
         if(button == PINB1)
            clear_bit(xdcp_cs_buffer,BIT1);
*/
     }
     // for channel 3
     if( diff_status & BIT_IMG6){
       if( old_status & BIT_IMG6 )
         set_bit(xdcp_cs_buffer,BIT2);
/*       if(temp_int_status & BIT_IMG6)
         if(button == PINB1)
           clear_bit(xdcp_cs_buffer,BIT2);
*/
     }
     // for channel 4
     if( diff_status & BIT_IMG7 ){
       if( old_status & BIT_IMG7 )
         set_bit(xdcp_cs_buffer,BIT3);
/*       if(temp_int_status & BIT_IMG7)
         if(button == PINB1)
           clear_bit(xdcp_cs_buffer,BIT3);
*/
     }
     old_status = temp_int_status;
}

/*****************************************************/
/* FlashLed()                                        */
/*                                                   */
/* output a pulse to the Port C pin 7 and            */
/* the LED will blink                                */
/*                                                   */
/* by   : zhaoqing Wang                              */
```

```
/* date  : 08/14/2001                                    */
/* updated :zhaoqing wang, 07/20/2003                    */
/*         Port B pin 3, because the Port C4-7 are       */
/*         outputs.                                      */
/*                                                       */
/*******************************************************/

void LightLed()
{
    output_low_port_b(BIT3);
}

void DarkLed()
{
    output_high_port_b(BIT3);
}

/*********************************************************************
**/
/* Pwm()
*/
/*
*/
/* Set capture/compare/PWM Regester1 for pulse width modulation(PWM)
*/
/* Refer to PIC16F87X manual p57 for details
*/
/*
*/
/* by  : zhaoqing wang
*/
/* date: 08/13/2001
*/
/*
*/
/*********************************************************************
**/

void Pwm()
{
    PR2 = PR2_P;
    CCPR1L = CCPR1L_P;
    set_bit( CCP1CON, BIT4);
    clear_bit( CCP1CON, BIT5);
    T2CON = T2CON_P;
    CCP1CON = CCP1CON_P;
}

/*******************************************************/
/* Init()                                              */
/*                                                     */
/* This function initializes the Microchip 16f876 system   */
/*                                                     */
/* By :   zhaoqing wang                                */
/* date :   08/11/2001                                 */
/*                                                     */
/*******************************************************/
```

```
void Init()
{

    ADCON1 = ADCON1_P;
    OPTION_REG = OPTION_P;
    PIE1 = PIE1_P;
    PIE2 = PIE2_P;
    PCON = PCON_P;
    T1CON = T1CON_P;

    set_tris_a( TRIS_A );       // set Port A as output.
    set_tris_b( TRIS_B );       // set port B <7:4> as input of
interrrupt.
                                // set port B <0:3> as output.
    output_port_b( 0x00 );
    set_tris_c( TRIS_C );       // set RC0 as input and others as
output.

    enable_interrupt( RBIE );   // enable RB port change interrupt
bit
    disable_interrupt( GIE );   // disable global interrupt bit
    interrupt_handle = 0;
    int_mem_offset = 0;
    old_status = 0xff;
    filt_status = 0xff;
    i = 0;
    j = 0;
    Pwm();
    eeaddress = XDCP_ADDR;    //read xdcp_ok from eeprom
    ReadByteFromEE();
    xdcp_ok = eedata;
    eeaddress = SPEED_ADDR;     // read falsh_speed from eeprom
    ReadByteFromEE();
    flash_speed = eedata;
    enable_interrupt( GIE);
}

/*************************************************************/
/* XdcpGoZero()                                              */
/* This function set the values of all potentiometers to 0. */
/*                                                           */
/*************************************************************/
void XdcpGoZero()
{
    output_port_a( 0x20 );  // output port A as 00100000
                            //enable the X9312(XDCP) Up/Down=DOWN
    delay_us(200);
    for  (k = 0; k < MAX_STEP; k++ )
    {
       output_low_port_a( BIT5 );
       nop();nop();nop();nop();nop();nop();nop();nop();nop();
       nop();nop();nop();nop();nop();nop();nop();nop();nop();
       output_high_port_a( BIT5 );
       nop();nop();nop();nop();nop();nop();nop();nop();
       nop();nop();nop();nop();nop();nop();nop();nop();
    }
```

```
    output_port_a( 0x3f );   //disable xdcp tune
    delay_us(100);
}
/*****************************************************************
*/
/* XdcpGoMax()
*/
/* This function set the values of all potentiometers to Max value.
*/
/* 07/03/2003, Zhaoqing Wang
*/
/*****************************************************************
*/
/*
void XdcpGoMax()
{
    output_port_a( 0x30 );  // output port A as 00110000
                            //enable the X9312(XDCP) Up/Down=Up

    for  (k = 0; k < MAX_STEP; k++ )
    {
       output_low_port_a( BIT5 );
       nop();nop();nop();nop();nop();nop();nop();nop();nop();
       nop();nop();nop();nop();nop();nop();nop();nop();nop();
       output_high_port_a( BIT5 );
       nop();nop();nop();nop();nop();nop();nop();nop();
       nop();nop();nop();nop();nop();nop();nop();nop();
    }
    output_port_a( 0x3f );   //disable and store the value of xdcp
}
*/
/*************************************************************/
/* XdcpGo1Step()                                             */
/*                                                           */
/* This function increments XDCP by one step.                */
/*                                                           */
/* XDPC must be enabled and set to up before calling this    */
/* function, and disabled after calling this function.       */
/*                                                           */
/* by  : zhaoqing wang                                       */
/* date : 08/14/2001                                         */
/* updated: zhaoqing Wang, 07/22/2003                        */
/*     This function is go one step. the pin4 of port A will */
/*     decide to increase or to decrease. Before calling this*/
/*     function, the PA4 shound be choosed                   */
/*                                                           */
/*************************************************************/
void XdcpGo1Step()
{
    output_low_port_a( BIT5 );
    nop();nop();nop();nop();nop(); //this keeps high level for a
certain time.
    nop();nop();nop();nop();nop();
    nop();nop();nop();nop();nop();
    output_high_port_a( BIT5 );
    nop();nop();nop();
    nop();nop();nop();nop();nop();
```

```
    nop();nop();nop();nop();nop();
}
void XdcpGo0()
{
    for(k=0; k<100; k++)
    {
        output_low_port_a( BIT5 );
        nop();nop();nop();nop();nop(); //this keeps high level for a
certain time.
        nop();nop();nop();nop();nop();
        nop();nop();nop();nop();nop();
        output_high_port_a( BIT5 );
        nop();nop();nop();
        nop();nop();nop();nop();nop();
        nop();nop();nop();nop();nop();
    }
}

/*****************************************************************/
/* XdcpGoHalf()                                                  */
/*                                                                 */
/* Adjust the petendiometer to the half of its maxiimum value  */
/*                                                                 */
/* by  : zhaoqing Wang                                           */
/* date : 08/14/2001                                             */
/*                                                                 */
/*****************************************************************/
/*
void XdcpGoHalf()
{
    output_port_a( 0x30 );  // output port_a as 00110000
                            //enable the X9312(XDCP) Up/Down=Up
                            //XDCP:X9312 Digitally Controlled
Potentiometer
    for( k=0; k< 50; k++ )
    {
       output_low_port_a(BIT5);
       nop();nop();nop();nop();nop();
       output_high_port_a(BIT5);
       nop();nop();nop();
    }
    output_port_a( 0x3f );   //disable xdcp tune
}
*/

/*****************************************************************
*/
/* DetectPeriod()
*/
/*
*/
/* This function detects pulse period. The period is obtained
*/
/* by averaging 8 successive periods. One period is obtained from
*/
/* 4 interrupts.
*/
```

```
/*****************************************************************
*/

/*
void DetectPeriod()
{
    int time_sum;
    period_int_num = 0;         // clear the times
    DETECT_INT = 1;           // set the detecting flag
    TMR1L = 0;                  // clear timer1
    enable_interrupt(INTE);
    enable_interrupt(GIE);
    while( TRUE )
    {
        if( period_int_num == 32)
        {
            disable_interrupt(INTE); //diable the RB0 interrupt
            break;
        }
    }
    DETECT_INT = 0;
    time_sum = period_time[1]-period_time[0];
    for ( k=1; k<=7; k++ )
    {
        time_sum = time_sum + (period_time[k] - period_time[k-1]);
    }
}
*/

/*********************************************************/
/* ReadTime()                                            */
/*                                                       */
/* This function is written by assembly to redueces      */
/* the delay betwwen reading tmr1l and tmr1h.            */
/* Even this delay is very short, TMR1H must be checked  */
/* after TMR1H is read to ensure that there is not change*/
/* of time when the lower byte was read.                 */
/*                                                       */
/*                                                       */
/* by : Zhaoqing Wang                                    */
/* date : 08/19/2001                                     */
/*********************************************************/
void ReadTime()
{
    asm {
        movf   TMR1H,W
        movwf  _read_tmr1+1
        movwf  _read_tmr1h
        movf   TMR1L,W
        movwf  _read_tmr1
        movwf  _read_tmr1l
        movf   TMR1H,W
        subwf  _read_tmr1+1,w
        btfsc  STATUS,Z
        return
        movf   TMR1L,W
        movwf  _read_tmr1
```

```
        movwf  _read_tmr1l
        movf   TMR1H,W
        movwf  _read_tmr1+1
        movwf  _read_tmr1h
        return
    }
}

/*****************************************************************/
/* Filter()                                                      */
/*                                                               */
/* This function filters out noise and unusal signals.           */
/*                                                               */
/* If the status of input keeps less than certain time,          */
/* The change of signal will be considered as noise or unusal    */
/* signal, and the output will remain unchanged.                 */
/*                                                          */
/* by  : zhaoqing wang                                           */
/* date : 08/19/2001                                             */
/* update: 08/27/2001                                            */
/*                                                               */
/*****************************************************************/

void Filter()
{
    if ( interrupt_handle > 0 )
    {
        interrupt_handle--;
      FiltIntHandling();
    }

    ReadTime();
    // for channel 1
    if ((filt_status & BIT_IMG0) == 0)
    {
        int_temp = read_tmr1-old_tmr11;
        if( int_temp > FILT_TIMEOUT )
        {
            if(old_status & BIT_IMG4)
                output_high_port_c( BIT4 );
            else
                output_low_port_c( BIT4 );
            set_bit(filt_status,BIT0);
        }
    }


    // for channel 2
    if ((filt_status & BIT_IMG1) == 0)
    {
        int_temp = read_tmr1-old_tmr12;
        if( int_temp > FILT_TIMEOUT )
        {
            if( (old_status & BIT_IMG5) == 0 )
                output_low_port_c( BIT5 );
            else
                output_high_port_c( BIT5 );
```

```
            set_bit(filt_status,BIT1);
        }
    }


    // for channel 3
    if ((filt_status & BIT_IMG2) == 0)
    {
        int_temp = read_tmr1-old_tmr13;
        if( int_temp > FILT_TIMEOUT )
        {
            if( (old_status & BIT_IMG6) == 0 )
                output_low_port_c( BIT6 );
            else
                output_high_port_c( BIT6 );
            set_bit(filt_status,BIT2);
        }
    }


    // for channel 4
    if ((filt_status & BIT_IMG3) == 0)
    {
        int_temp = read_tmr1-old_tmr14;
        if( int_temp > FILT_TIMEOUT )
        {
            if( (old_status & BIT_IMG7) == 0 )
                output_low_port_c( BIT7 );
            else
                output_high_port_c( BIT7 );
            set_bit(filt_status,BIT3);
        }
    }


}

/*********************************************/
/*  function name: C2Int()                   */
/*      it converts 2 char to 1 int          */
/*  input:    char: read_tmr1l               */
/*                  read_tmr1h               */
/*  output:    int: temp_int_time            */
/*                                           */
/*  by :  zhaoqing wang                      */
/* date:  08/28/2001                         */
/*                                           */
/*********************************************/

void C2Int()
{
 asm{
      movf  _read_tmr1l,W;
      movwf _temp_int_time;
      movf  _read_tmr1h,W;
      movwf _temp_int_time+1;
    }
```

```
}

/*****************************************************/
/*  function name: FiltIntHanding()                  */
/*                                                   */
/*  by  : Zhaoqing Wang                              */
/* date : 08/28/2001                                 */
/* update from the laser.prj   (inter.c)             */
/*                                                   */
/*****************************************************/

void FiltIntHandling()
{

     char temp_int_status;

     read_tmr1l = interrupt_time_status[i];
     i++;
     read_tmr1h = interrupt_time_status[i];
     i++;
     temp_int_status = interrupt_time_status[i];
     i++;
     C2Int();
     if( i >= INT_MAX_NUM)
         i = 0;
     diff_status = old_status ^ temp_int_status;
     old_status = temp_int_status;

     // channel 1
     if( diff_status & BIT_IMG4 )
     {
         old_tmr11 = temp_int_time;
         clear_bit(filt_status,BIT0);
     }

     // channel 2
     if( diff_status & BIT_IMG5 )
     {
         old_tmr12 = temp_int_time;
         clear_bit(filt_status,BIT1);
     }

     // channel 3
     if( diff_status & BIT_IMG6 )
     {
         old_tmr13 = temp_int_time;
         clear_bit(filt_status,BIT2);
      }

     // channel 4
     if( diff_status & BIT_IMG7 )
     {
         old_tmr14 = temp_int_time;
         clear_bit(filt_status,BIT3);
     }

}
```

```
/*********************************************************/
/* Reads one byte from the EEprom at the specified address */
/* and returns it                                          */
/*********************************************************/
void ReadByteFromEE()
{
    asm{
        bcf    STATUS,RP1      ;bank0
        movf   _eeaddress,W      ; write address
        bsf    STATUS,RP1
        bcf    STATUS,RP0      ;BANK2
        movwf  EEADR           ;Read from this address

        bsf    STATUS, RP0        ;bank3
        bcf    EECON1, EEPGD      ;Point to EE memory
        bsf    EECON1, RD         ;Initiate a read cycle

        bcf    STATUS,RP0
        movf   EEDATA,W           ;Fetch byte from dataregister
        bcf    STATUS,RP1
        bcf    STATUS,RP0
        movwf  _eedata
    }
}

/*********************************************************/
/* Writes one byte to the EEprom at the specified address */
/*********************************************************/
void WriteByteToEE()
{
    asm{
        bcf    STATUS,RP1
        bcf    STATUS,RP0     ; bank0
        movf   _eedata,W      ; Data to write
        bsf    STATUS,RP1     ; bank 2
        movwf  EEDATA
        bcf    STATUS,RP1     ;bank0
        movf   _eeaddress,W   ;Address to write to
        bsf    STATUS,RP1     ; bank 2
        movwf  EEADR

        bsf    STATUS,RP0     ; bank3
        ;label1
        ;btfsc  EECON1,WR      ;wait for write to complete
        ;goto   label1

      bsf    EECON1,WREN    ;Enable writes to the EEProm
        bsf    STATUS,RP0
        bcf    STATUS,RP1     ;bank1
      bcf    INTCON,GIE     ;Disable interrupts during write

        bsf    STATUS,RP1     ;bank3
        movlw  0x55
        movwf  EECON2         ;write 55h to EECON2
        movlw  0xAA
        movwf  EECON2         ;Initiate a write cycle
```

```
      bsf    EECON1,WR      ;start write operation
      bcf    EECON1,WREN     ;       // Disable writes to EEProm
      bsf    INTCON,GIE     ;set interrupts enable
    }
}


/*****************************************************/
/* the main program only for test                    */
/*                                                   */
/*    By : zhaoqing Wang                             */
/*   date: 08/11/2001                                */
/*   waiting interrup and output portc<2> circlely   */
/*                       1<--->0                     */
/*****************************************************/

void main()
{
    delay_s(5);         // if system comes to steady, it start to
work.
    Init();
    j = 0;

    enable_interrupt(RBIE);
    enable_interrupt(GIE);
    old_tmr11 = 0;
    old_tmr12 = 0;
    old_tmr13 = 0;
    old_tmr14 = 0;
    TMR1 = 0;
    bc0=0;
    bc1=0;
    bc2=0;
    bc3=0;
    while ( TRUE )
    {
        if(input_pin_port_b( BIT1 ) == LOW_LEVEL )
        {
           delay_s(1);
           if(input_pin_port_b( BIT1 ) == LOW_LEVEL )
           {
               button = PINB1;
               XdcpSet();
               button = NO_BT;
           }
        }
        else{
        // for test skip XdcpSet(), only run the Filter()
          if (input_pin_port_c( BIT0 ) == LOW_LEVEL )
          {
            delay_s(1); //eliminated trembling
            if (input_pin_port_c( BIT0 ) == LOW_LEVEL )
            {
                button = PINC0;
                //output_low_port_b(BIT3);
                XdcpSet();    // adjust XDCP
```

```
                disable_interrupt( GIE );
                temp_port = input_port_b(); // output of portb<0:3>
are same as portb<4:7>
                output_port_c( temp_port );
                eeaddress = XDCP_ADDR;
                eedata = xdcp_ok;
                WriteByteToEE();   // write the status into flash
mem
                eeaddress = SPEED_ADDR;
                eedata = flash_speed;
                WriteByteToEE();
                old_tmr11 = 0;
                old_tmr12 = 0;
                old_tmr13 = 0;
                old_tmr14 = 0;
                filt_status = 0x00;
                TMR1 = 0;
                interrupt_handle = 0; //init the interrupt
                int_mem_offset = 0;
                i = 0;
                old_status = input_port_b();
                enable_interrupt( GIE );
                button = NO_BT;
            }
          }
          else{

            /* for no filter through high 4 bits of B port out to
high 4 bits of C port */

            temp_port = input_port_b();
            output_port_c( temp_port );

            Filter();
        }
        }
      switch( xdcp_ok )
        {
          case XDCP_ERR:
              int_temp = 100 * flash_speed;
              if (j <=  int_temp)
                  LightLed();
              else
              {
                  DarkLed();
                  int_temp = 100*(flash_speed+flash_speed);
                  if( j >= int_temp )
                     j = 0;
              }
              j++;
          case XDCP_WORK:
              output_low_port_b(BIT3); //LightLed();
          default:
              output_high_port_b(BIT3); //DarkLed();
        }
    }
}
```

```
/** End of File **/
```

Source Code of device driver in Linux

DIO96.h

```
/*++
 * Module Name:
 *
 *    DIO96.h
 *
 * Abstract:
 *
 *    Define National Instruments PC-DIO-96 board address and
registers.
 *
*
 * Environment:
 *
 *    RTX application.
 *
 * Revision History:
 *
--*/

//#define IRQ           0x05  // Interrupt vertor for NI
//#define BASE_ADDR     0x180 // Base address of the NI
//#define PTL(x)  (*(PLONG)(x))
/* PPI A */

#define VENDOR_NI        0x1093
#define DIO96_ID         0x0160
#define BUFSIZE          400
//#define HALF_BUFFER_SIZE BUFFER_SIZE/2

#define BASE_ADDR     baseptr1
#define PORTA_ADDR      BASE_ADDR + 0x00 // Port A, aka. Port 0
#define PORTB_ADDR      BASE_ADDR + 0x01 // Port B, aka. Port 1
#define PORTC_ADDR      BASE_ADDR + 0x02 // Port C, aka. Port 2
#define CNFG_ADDR BASE_ADDR + 0x03 // Config Register

/* PPI B */
#define B_PORTA_ADDR    BASE_ADDR + 0x04 // PPI B Port A, aka. Port
3
#define B_PORTB_ADDR    BASE_ADDR + 0x05 // PPI B Port B, aka. Port
4
#define B_PORTC_ADDR    BASE_ADDR + 0x06 // PPI B Port C, aka. Port
5
#define B_CNFG_ADDR     BASE_ADDR + 0x07 // PPI B Config Port

/* PPI C */
```

```
#define C_PORTA_ADDR    BASE_ADDR + 0x08 // PPI C Port A, aka. Port
6
#define C_PORTB_ADDR    BASE_ADDR + 0x09 // PPI C Port B, aka. Port
7
#define C_PORTC_ADDR    BASE_ADDR + 0x0A // PPI C Port C, aka. Port
8
#define C_CNFG_ADDR     BASE_ADDR + 0x0B // PPI C Config Port

/* PPI D */
#define D_PORTB_ADDR    BASE_ADDR + 0x0D // PPI D Port B, aka. Port
10
#define D_PORTC_ADDR    BASE_ADDR + 0x0E // PPI D Port C, aka. Port
11
#define D_CNFG_ADDR     BASE_ADDR + 0x0F // PPI D Config Port

/* Counter/Timer */
#define CLOCKA          BASE_ADDR + 0x10 // Clock or Counter 0
#define CLOCKB          BASE_ADDR + 0x11 // Clock or Counter 1
#define CLOCKC          BASE_ADDR + 0x12 // Clock or Counter 2
#define CLOCK_CTRL      BASE_ADDR + 0x13 // Clock or Counter Control

/* Interrupt control */
#define INTR_CTRL1      BASE_ADDR + 0x14 // First interrupt control
reg
#define INTR_CTRL2      BASE_ADDR + 0x15 // Second interrupt control
reg


//#define STOP_TIME     5     // Minutes for shutdown handlers to
wait after
                        // a stop - 0 for indefinite.


/*
typedef struct {
//    LONG  Pid;        // Process ID of sender
      LONG    BufFull;
      LONG  Ack;        // Server acknowledge flag
    UCHAR   Buffer[BUFFER_SIZE];
} MSGSTR, *PMSGSTR;

*/
```

---

control.h

```
#define DATA1 0
#define DATA2 1
#define COMMAND_FIFO 2
#define TASK_CONTROL_FIFO 3
#define SEMEPHORE_FIFO 4

#define START_TASK      1
#define STOP_TASK 2
#define TICK_RESOLUTION 1e6
#define DIO96_CONF      0x82
```

```
#define FIFO_0_READY    1
#define FIFO_1_READY    0

struct msg_struct {
      short int command;
      unsigned char conf;
//    int task;
      int period;
      int buffer_size;
};
```

dio96_module.c

```
/*****************************************************************
 * dio96_module.c
 * Drive driver for DIO-96 digital I/O board in RTLinux
 * Last modified: 05/28/2001

*****************************************************************/
#include <linux/stddef.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/pci.h>
#include <linux/config.h>
#include <linux/vmalloc.h>
#include <asm-i386/io.h>
#include <linux/errno.h>
#include <rtl.h>
#include <rtl_sched.h>   /*time.h is included in this file*/
#include <rtl_fifo.h>
#include "DIO96.h"
#include "control.h"

/* disable usage counter */
#undef MOD_INC_USE_COUNT
#undef MOD_DEC_USE_COUNT
#define MOD_INC_USE_COUNT
#define MOD_DEC_USE_COUNT

unsigned char * baseptr0;
unsigned char * baseptr1;
int status[2] = {1,1};
pthread_t *dio96_task;
struct msg_struct msg;
int i;

unsigned long dio96_setup(void);
void *thread_code(void *t);
int msg_handler(unsigned int fifo);
int data_handler(unsigned int fifo);

int init_module(void) {
    int c[5];
    pthread_attr_t attr;
    struct sched_param sched_param;
    int ret;
```

```
    /* configure dio96*/
    *baseptr1 = dio96_setup();

    /* create fufo */
    rtf_destroy(0);
    rtf_destroy(1);
    rtf_destroy(2);
//    rtf_destroy(3);
    rtf_destroy(4);
    c[0] = rtf_create(0, BUFSIZE);
    c[1] = rtf_create(1, BUFSIZE);
    c[2] = rtf_create(2, 40);
//    c[3] = rtf_create(3, 20);
    c[4] = rtf_create(4, 20);
    printk("Fifo return 0=%d 1=%d 2=%d 3=%d
4=%d\n",c[0],c[1],c[2],c[3],c[4]);

    /* create a real-time task*/
    dio96_task = vmalloc(sizeof(pthread_t));
    pthread_attr_init (&attr);
    sched_param.sched_priority = 4;
    pthread_attr_setschedparam (&attr, &sched_param);
    ret = pthread_create (dio96_task,  &attr, thread_code, (void
*)1);

    /* create fifo handlers */
    rtf_create_handler(0, &data_handler);
    rtf_create_handler(1, &data_handler);
    rtf_create_handler(2, &msg_handler);

    return 0;
}

/*Kernel module thread*/
void *thread_code(void *t) {
    int count = 0;
    short int rtf = 0;
    short int ret;
    int n;
/*
    int i;
    i=0;
*/
    while(1) {
        ret = pthread_wait_np();
        /* check if the FIFO is filled up*/
        if(count == msg.buffer_size) {
            count  = 0;
            status[rtf] = 0;
            n = rtf_put(4, &rtf, 1);
            rtf=(~rtf)&0x01;
        }
//        if(status[rtf] == 0)
//            printk("queue overflow\n");

        /* write data into the FIFO*/
        rtf_put(rtf, baseptr1, 1);
```

```
/*
        if (*baseptr1 != 0xff && *baseptr1 != 0x0)
        {
            printk("%0x ", *baseptr1);
            i++;
            if( i == 16 ){
                printk("\n");
                i=0;
            }
        }
*/
        count++;

    }
    return 0;
}

/* handler for message FIFO*/
int msg_handler(unsigned int fifo)
{
    if ((rtf_get(COMMAND_FIFO, &msg, sizeof(msg))) == sizeof(msg)) {
        rtl_printf("Task: executing the command to task; command:%d
period:%d; \
                    buffer_size: %d\n",msg.command, msg.period,
msg.buffer_size);
            switch(msg.command) {
                case START_TASK:                    /* Start data
reading thread */
                    writeb(msg.conf, CNFG_ADDR);
                    printk("START_TASK \n");
                    pthread_make_periodic_np(*dio96_task,
gethrtime(), msg.period);
                    pthread_wakeup_np(*dio96_task);
                    break;
                case STOP_TASK:                     /* Suspend data
reading thread */
                    printk("STOP_TASK \n");
                    pthread_suspend_np(*dio96_task);
                    break;
                default:
                    rtl_printf("RTL task: bad command\n");
                return 0;
            }
    }
    return 0;
}

/*handler for data FIFOs*/
int data_handler(unsigned int fifo)
{
#ifdef DEBUG
    printk("fifo = %d\n", fifo);
#endif
    status[fifo] = 1;
    return 0;
}
```

```
/* detect and config DIO_96 board*/
unsigned long dio96_setup(void) {
    #ifdef CONFIG_PCI
    if(pcibios_present()){
//        unsigned long base_addr1, base_addr0, pci_irq_line,
offset;
        u32 base_addr1, base_addr0, pci_irq_line, offset,
window_value;
        u8 pci_cmd;
        struct pci_dev *dev = NULL;

      /* find PCI-DIO-96 in PCI slots */
        dev=pci_find_device(VENDOR_NI, DIO96_ID, dev);

        /* Disable Master/IO access, Enable memory access */
        pci_set_master(dev);
        pci_read_config_byte(dev, PCI_COMMAND, &pci_cmd);
        pci_cmd |= PCI_COMMAND_MEMORY;
        pci_cmd &= ~PCI_COMMAND_IO;
        pci_cmd |= PCI_COMMAND_IO;
        pci_cmd |= PCI_COMMAND_MASTER;
        pci_cmd |= PCI_COMMAND_INVALIDATE;
        pci_write_config_byte(dev, PCI_COMMAND, pci_cmd);

        /* get base addresses for Mite and I/O register and IRQ line
of board*/
        base_addr1 = dev->base_address[1];
        base_addr0 = dev->base_address[0];
        pci_irq_line = dev->irq;
#ifdef DEBUG
        printk("IRQ = %d\n", pci_irq_line);
#endif

        /* remap base addresses to virtual memory space*/
        offset = base_addr1 & ~PAGE_MASK;
        base_addr1 &= PCI_BASE_ADDRESS_MEM_MASK;
        base_addr1 &= PAGE_MASK;
        baseptr0 = ioremap(base_addr0, 1024*4);
        baseptr1 = ioremap(base_addr1, 1024*4);
//#ifdef DEBUG
        printk("offset = 0x%x\n", offset);
        printk("base_addr_after_mask = 0x%x\n", base_addr1);
        printk("baseptr0 = 0x%x\n", baseptr0);
        printk("baseptr1 = 0x%x\n", baseptr1);
//#endif

        /* Configure Mite */
        /* the window value should calculated from the base addr
before remapping*/
        window_value = (0xffffff00 & base_addr1) | 0x00000080;
        writel(0x0000aeae, (baseptr0+0x0340));
        writel(window_value, (baseptr0+0x00c0));

        /* Testing: write to the offset 0 of the area */
        writeb(0x80,(baseptr1+0x03));
        writeb(0x00,baseptr1);
```

```
#ifdef DEBUG
        printk("*baseptr0 = 0x%x\n", readb(baseptr0));
        printk("*(baseptr0+3) = 0x%x\n", readb(baseptr0+0x03));
        printk("*baseptr1 = 0x%x\n", readb(baseptr1));
        printk("*(baseptr1+3) = 0x%x\n", readb(baseptr1+0x03));
#endif
    }
    else {
        printk("No PCI board detected");
        return -ENODEV;
    }
    #endif
    return *baseptr1;
}

void cleanup_module(void) {
#ifdef DEBUG
    printk("%d\n", rtf_destroy(1));
    printk("%d\n", rtf_destroy(2));
#else
    rtf_destroy(1);
    rtf_destroy(2);
    rtf_destroy(3);
    rtf_destroy(4);
    rtf_destroy(5);
#endif

    /* unmap when we unload the driver */
    iounmap(baseptr0);
    iounmap(baseptr1);

    pthread_delete_np (*dio96_task);
    vfree(dio96_task);

    printk("bye dio96...\n");
    return;
}

/** End of File **/
```