Modeling and control of the Pr-500-Fl pavement profiler cutter system

James William Porrazzo
University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation
https://digitalscholarship.unlv.edu/rtds/423

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
Modeling and Control of the
PR-500-FL Pavement Profiler
Cutter System

by

James W. Porrazzo

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science

in

Mechanical Engineering

Department of Mechanical Engineering
University of Nevada, Las Vegas
December, 1994

Developed under DOE Grant: DOE-UNLV 93-01
The thesis of James W. Porrazzo for the degree of Master of Science in Mechanical Engineering is approved.

Georg F. Mauer Ph.D.
Chairperson.

Robert E. Skaggs Ph.D.
Examiner Committee Member.

Douglas D. Reynolds Ph.D.
Examiner Committee Member.

Evangelos A. Yfantis Ph.D.
Graduate Faculty Representative.

Ronald W. Smith Ph.D.
Graduate Dean.

University of Nevada, Las Vegas
December, 1994
Abstract

A manually controlled PR-500-FL Pavement Profiler vehicle is modeled and converted to allow terrain contingent cutter depth control by an on-board portable computer. A series of geometric equations were developed in an attempt to define the required motion of the vehicle's three hydraulic actuator legs. A geometric model was used to create a MATRIXX simulation verifying the hydraulic leg motions and cutter depth control. Ultrasonic sensors are used to acquire terrain profile data. Linear displacement transducers are used to measure the vertical hydraulic leg displacements required for cutter depth adjustments. A dual axis angular tilt sensor was installed as an additional vehicle safety feature. A description of the design and installation of the complete sensor and control system is presented. A number of experimental runs were performed, and the results acquired from the installed system are evaluated. Recommendations for improvements of the current system, and possible future systems are presented.
## Contents

Abstract ................................................................. iii
List of Figures ........................................................... vi
Acknowledgements ...................................................... vii

1 Introduction ................................................................. 1
   1.1 Purpose and Scope of Thesis ................................ 1
   1.2 Organization of Thesis ........................................ 3
   1.3 Literature Search ................................................ 4

2 Geometric Model ......................................................... 5
   2.1 Geometric Modeling .............................................. 7
   2.2 Vehicle Rotation about the X-Y-Z axis .................... 8

3 MATRIXx Model .......................................................... 12
   3.1 Side, Front and Cutter Model ............................... 13
   3.2 Spatial Distribution of Terrain Elevation ................. 16
   3.3 Integrated Model ................................................ 18
   3.4 MATRIXx Command File .................................... 20
   3.5 Simulation Results .............................................. 20

4 Hardware Design ......................................................... 23
   4.1 Sensor Configurations .......................................... 23
      4.1.1 Ultrasonic Sensors ..................................... 25
      4.1.2 LDT Sensors ............................................ 28
      4.1.3 Tilt Sensor .............................................. 32
   4.2 Console Cabinet Layout ....................................... 32
      4.2.1 Computer and Data Acquisition ...................... 35
      4.2.2 Power and Relay Configuration ...................... 37

5 Software Design .......................................................... 39
   5.1 Program Organization .......................................... 39
   5.2 Computer Interface ............................................ 44
6 Experimental Results

7 Conclusions

7.1 Conclusions ........................................... 53
7.2 Recommendations ................................. 54
Bibliography ............................................. 54
Appendixes .............................................. 56
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Isometric View of a basic PR-500-FL Pavement Profiler</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>2-D Geometric Model of Pavement Profiler</td>
<td>7</td>
</tr>
<tr>
<td>2.3</td>
<td>Series View of Side Model vs. Terrain Movement</td>
<td>8</td>
</tr>
<tr>
<td>2.4</td>
<td>X-Y-Z Plane Geometry</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Profiler Critical Point Definition</td>
<td>13</td>
</tr>
<tr>
<td>3.2</td>
<td>Hydraulic Actuator Position Control</td>
<td>14</td>
</tr>
<tr>
<td>3.3</td>
<td>Typical Elevator Control Model</td>
<td>14</td>
</tr>
<tr>
<td>3.4</td>
<td>Cutter/Elevator Control Model</td>
<td>15</td>
</tr>
<tr>
<td>3.5</td>
<td>Vertical Signal Correction</td>
<td>16</td>
</tr>
<tr>
<td>3.6</td>
<td>Discrete Time Delay</td>
<td>17</td>
</tr>
<tr>
<td>3.7</td>
<td>Spatial Propagation Model</td>
<td>18</td>
</tr>
<tr>
<td>3.8</td>
<td>MATRIX(x) Complete Control Model</td>
<td>19</td>
</tr>
<tr>
<td>3.9</td>
<td>MATRIX(x) Control Simulation Results</td>
<td>22</td>
</tr>
<tr>
<td>4.1</td>
<td>Sensor Locations on the PR-500-FL Pavement Profiler</td>
<td>25</td>
</tr>
<tr>
<td>4.2</td>
<td>Table of DCU-7e-8 Ultrasonic Sensor Specifications</td>
<td>28</td>
</tr>
<tr>
<td>4.3</td>
<td>Table of Linear Displacement Transformer Specifications</td>
<td>29</td>
</tr>
<tr>
<td>4.4</td>
<td>Linear Displacement Transformer Housing Design</td>
<td>31</td>
</tr>
<tr>
<td>4.5</td>
<td>Computer and Electrical Cabinet Diagram</td>
<td>34</td>
</tr>
<tr>
<td>4.6</td>
<td>Real Time Input Channel Definitions</td>
<td>36</td>
</tr>
<tr>
<td>4.7</td>
<td>Digital Output Channel Definitions</td>
<td>37</td>
</tr>
<tr>
<td>5.1</td>
<td>Profiler Computer Control Screen</td>
<td>43</td>
</tr>
<tr>
<td>5.2</td>
<td>Profiler Computer Control Schematic</td>
<td>44</td>
</tr>
<tr>
<td>5.3</td>
<td>Profiler Computer Control Flowchart</td>
<td>45</td>
</tr>
<tr>
<td>6.1</td>
<td>1 inch Cut Zero Start Depth</td>
<td>48</td>
</tr>
<tr>
<td>6.2</td>
<td>2 inch Cut Zero Start Depth</td>
<td>49</td>
</tr>
<tr>
<td>6.3</td>
<td>3 inch Cut Zero Start Depth</td>
<td>50</td>
</tr>
<tr>
<td>6.4</td>
<td>4 inch Cutting Depth after Leveling</td>
<td>52</td>
</tr>
</tbody>
</table>
Acknowledgments

First and foremost I would like to dedicate this thesis to my mother and father, Marguerite and Joseph, for their encouragement and financial support without which I would never have continued my education. I would like to thank Dr. Georg Mauer Ph.D. for my education in automatic control and mechanical engineering and also for his invaluable support and guidance throughout this project. I would like to thank Dr. Skaggs, Dr. Reynolds, and Dr. Yfantis for their willingness to sit on my thesis committee. A special thanks to Glen Hein for his work on the projects computer control program and to Andy Bagdanov for his assistance in my program understanding. My sincerest gratitude to all the professors and staff of the mechanical engineering department of UNLV for educating me and in the caring and considerate environment that they have provided for students over the years.
Chapter 1

Introduction

1.1 Purpose and Scope of Thesis

The Desert Research Institute was commissioned by the Department of Energy to develop a means by which surface contaminants with radionuclides, located at the Nevada Test Site, could be efficiently removed, transported, and processed for contaminant removal. The DOE selected the PR-500-FL Pavement Profiler (with modification) as the vehicle for soil removal. Dr. Georg Mauer, Associate Professor of the Mechanical Engineering Department of the University of Nevada, Las Vegas, was retained to investigate and modify the vehicle so that it would satisfactorily perform this function. The soil removal depth of the PR-500-FL Pavement Profiler is adjustable. It is desired to minimize the amount of radioactive soil collected by continuously adjusting the cutting depth in accordance with the depth of radio-nuclide penetration. In order to implement this, the Pavement Profiler must be able to accurately track the radioactive terrain and continually adjust its cut-
ter depth accordingly. In its factory configuration the Pavement Profiler is designed for frequent adjustment of cutting depth.

The purpose of this thesis is the modeling and real time control of the PR-500-FL Pavement Profiler in an off-road environment. The PR-500-FL Pavement Profiler was originally designed by the Caterpillar company to excavate roadbeds in preparation for later resurfacing. The profiler cuts a specific depth of paving material off of the existing road surface. Under its original design, the profiler cutting depth is intended to remain unaltered. The PR-300-FL Pavement Profiler was never designed to be operated off-road.

Operation of the profiler in an off-road environment causes a number of problems that were not addressed in the original design concept by the Caterpillar company. If the cutter depth were to be constantly maintained at a set depth the profiler would remove the top portion of any high areas that it traversed while leaving low areas untouched. This is unacceptable profiling because those areas untouched by the profiler would retain radioactive material while those areas actively profiled would result in an excess of removed material. When used on an uneven off-road environment, constant rapid readjustment of the cutter depth is required. The required rapid readjustment of the cutter depth cannot be adequately implemented by the operator and still allow proper safe directional control of the vehicle.

The scope of this project is the conversion of the PR-500-FL Pavement
Profiler so that it will automatically adjust the cutting depth to the level of radio-nuclide contamination while tracking the terrain contour. The control of the cutting depth of the PR-500-FL Pavement Profiler will be delegated to an on-board computer system. An accurate geometric and simulation model of the vehicle system will be needed for control system design. Among the information needed for computer model generation are: vehicle geometry, system configuration, spatial (terrain) and temporal signal propagation and hydraulic system dynamics. After the generation and validation of the computer model has been completed, means of implementing the control algorithm will be investigated, implemented, and tested.

The X.Y.Z coordinates in Fig. 2.1 denote a Cartesian 3-D frame of reference used throughout this thesis. The x-axis denotes the direction of forward motion, the y-axis denotes the plane of the horizontal coordinates perpendicular to the x-axis, and the z-axis is defined vertically upward.

1.2 Organization of Thesis

The thesis organization follows the normal development of the project. The first step in the project development was the creation of a geometric vehicle model and the determination of the geometric equations associated with that model. Next, the model was used as the basis for a simulation with the MATRIXX language. After debugging and validation of the MATRIXX model, the control algorithm was applied to the real world environment of the
vehicle. Through the use of an on-board computer and the integration with other associated hardware, real time control of the vehicle was accomplished.

1.3 Literature Search

Few literature sources exist relating to the control of Pavement Profilers or other heavy construction equipment. Little research into the automation of heavy construction machinery has taken place. This is probably due to the diversity of tasks that the various pieces of equipment are required to perform. A human operator is always required on such equipment because of the complexity of tasks, and for safety reasons, therefore little automation has been accomplished. A limited amount of published data pertaining to the Pavement Profiler is available [2]. The theory behind the control of the vehicle was developed using a number of standard control text. [14][4][13][12][6]. In the construction of the MATRIXx model, Bahram Shahian's book [15] was extremely useful along with the documentation provided with the MATRIXx software[7]. Several articles [8] [9] [10] have been published dealing with autonomous control of various hostile terrain vehicles and were helpful in determining the viability of various environment sensing methods. In the area of transforming the model into its real-time configuration a variety of texts were used, which are listed in later chapters.
Chapter 2

Geometric Model

The standard (non-modified) PR-500-FL Pavement Profiler is shown in Fig. 2.1 along with its basic dimensions and inherent dimensional constants listed below:

Figure 2.1: Isometric View of a basic PR-500-FL Pavement Profiler
• Body length: 25' 11"  Body width: 8' 2.5"  Height: 9' 10"

• WB: The front to rear track center distance = 216 inches.

• CR: The cutter to rear track center distance = 95 inches.

• FC: The cutter to front track center distance = 121 inches.

• LR: The left front to right front track center distance = 66 inches.

• FS: The front track center to sensor distance = 72 inches

The PR-500-FL Pavement Profiler rides on three tracks, two in the front and one in the center rear. Each track has a vertically variable hydraulic suspension which is used to control the cutter depth. By varying the height of the tracks individually the depth and amount of soil actually removed by the cutter can be closely controlled. The time constants for the hydraulic actuators are approximately 5 inch per second. Using this information we can approximate the response of the actuators during the computer simulation portion of the project. The actual actuator response time will be fine tuned later during the vehicle modification phase of this project. By simulating the vehicle geometrically the varying terrain soil removal is minimized and the cutter depth errors will be greatly reduced.
2.1 Geometric Modeling

Fig. 2.2 represents a 2-dimensional model of the pavement profiler vehicle in the x-z plane through the geometrical center. For simplicity of modeling, the front right and front left sides of the vehicle are assumed to be symmetrical and subjected to the same terrain variations.

By simulating the motion of the 2-D model across varying terrain with stationary elevators, we can better understand the vertical motion. From Fig. 2.3 we can see that the three critical positions of the profiler model (front track, cutter, and rear track) all vary interdependently and also dependent upon the terrain beneath them. As the vehicle moves over the terrain elevation variations, the effects on the cutting depth are a function of the spatial distribution of elevations at the vehicle front, cutter, and rear.
2.2 Vehicle Rotation about the X-Y-Z axis

When the terrain elevations at the right and left front tracks differ from each other, the elevation at the vehicle center (y-z plane) is found as follows:

Let RFT and LFT denote the terrain elevations at the right and left front tracks, respectively, in the front view of Fig 2.4. The elevation at the geometric center front is found as the arithmetic mean.

When the terrain elevations at the left front and rear center tracks differ from each other, the elevation at the vehicle center (x-z plane) is found as follows:

Let LFT and RT denote the terrain elevations at the right and left front tracks, respectively, in the side view of Fig 2.4. The elevation at the
center front is found as the sine length of similar triangles.

Figure 2.4: X-Y-Z Plane Geometry

The definitions for the variables depicted in Fig. 2.4 are listed below.

- **CAD**: Cutter Actual Depth
- **CDD**: Cutter Desired Depth
- CR: Cutter to Rear track center distance = 95"
- FC: Front track center to Cutter distance = 121"
- WB: Front track center to Rear track center distance = 216"
- LR: Left Front to Right Front distance
- RFE: Right Front Elevation measurement.
- RFT: Right Front Terrain measurement.
- LFE: Left Front Elevation measurement.
- LFT: Left Front Terrain measurement.
- RE: Rear Elevation measurement.
- RT: Rear Terrain measurement.

Using the standard geometric relationship between similar triangles, the following equation can be written:

\[
CAD = \frac{(RT - RE) \times FC}{WB} + \frac{(LFT - LFE + RFT - RFE) \times CR}{(2 \times WB)}
\]  

(2.1)

By substituting the known desired cutter depth (CDD) for the actual cutter depth (CAD) in equation 2.1, then rearranging and backing out the desired variables (LFE, RFE, and RE) yields three separate equations:
These three equations, 2.2, 2.3, and 2.4 define the elevation of the hydraulic actuator on each of the vehicle's three legs. The input values for the model equations 2.1-2.4 are obtained from on-vehicle measurement. Using the geometric equations and the MATRIX\(\_\) control simulation software package, a simulation model will be constructed in the next chapter.
Chapter 3

MATRIX$_X$ Model

Having defined the variables and parameters of the PR-500 Pavement profiler and having modeled the geometry we proceed with a simulation of the vehicles dynamic processes. There are many candidate design and analysis software packages available currently. The MATRIX$_X$ software package was selected for analysis and simulation of our control problem because it provides a graphically programmable problem solving environment with a block diagram type graphical user interface. MATRIX$_X$ is a programmable, numerical matrix solving environment, with graphic user interface, which can solve the complex matrix problems encountered in a control engineering environment. MATRIX$_X$ was designed specifically for the solution of complex classical and modern control problems.

From documentation on the PR-500-FL Pavement Profiler a geometric model was developed. The MATRIX$_X$ simulation model is built from a number of lower order systems. When building the simulation model, the
distance from the front ultrasonic sensors to several critical points on the vehicle must be taken into consideration. Those critical points are shown below in Fig. 3.1.

![Figure 3.1: Profiler Critical Point Definition](image)

3.1 Side, Front and Cutter Model

The MATRIXX model for a hydraulic elevator uses the general algebraic expression SystemBuild™ block for the geometry modeling equations. 2.1, 2.2, 2.3, and 2.4. A data switching block allows access to the terrain data at the respective location of the part relative to the terrain elevator measurement at the front. A linearized discrete feedback loop with limited integration simulates the hydraulic actuator position control. and is
shown in Fig. 3.2.

![Figure 3.2: Hydraulic Actuator Position Control](image)

The assembled model of a typical elevator control system is shown in Fig. 3.3 below.

![Figure 3.3: Typical Elevator Control Model](image)

The elevator control model is used for each of the three legs. A modular model for the cutter elevation control is constructed by combining all three leg models. The resulting MATRIX model is shown in Fig. 3.4.
Figure 3.4: Cutter/Elevator Control Model
3.2 Spatial Distribution of Terrain Elevation

In order to acquire an accurate elevation profile of the terrain the ultrasonic sensor signals have to be stored and accessed in accordance with the vehicle's travel distance. The ultrasonic sensors are located 72 inches above ground and 72 inches in front of the center of each front track. As seen in Figs. 2.2 and 2.3, the elevator signal requires a correction for the elevation of the sensor above the X-Y-axis. The equations describing this correction for both the left and right sensors are shown in Fig. 3.5.

![Figure 3.5: Vertical Signal Correction](image)

When sampling every inch of vehicle forward travel, the front elevation is given by the sample taken 72 periods prior to the current sample. This sampling shift takes place as each critical point is addressed. The front center elevation is calculated as the arithmetic mean between the front left and right
elevators. The discrete time delay blocks for the front right, front left, rear, and cutter critical points are shown in Fig. 3.6. The simulation increments at a 1 inch step size and the sampling period is arbitrarily chosen as 1 second at a vehicle forward velocity of 1 in./sec.

Using a similar approach as described in Fig 3.3 the terrain signals at each critical vehicle point are determined. The vehicle travel distance is determined by integrating the vehicle velocity. The three spatial shifts of the terrain signal and the velocity simulation are connected together to generate the terrain profile for each of the vehicle's critical points. Fig. 3.7 shows the combined model.
3.3 Integrated Model

At this final stage of model development, the shift sequencing and the cutter elevation control are combined to complete the simulation model. There are two inputs to the simulation in the form of sine waves. Using sine waves provides a diverse input simulating a continuously changing terrain profile. All other variables in the simulation are initially set to zero or calibration conditions. Fig. 3.8 shows the completed pavement profiler control simulation model.
Figure 3.8: MATRIXN Complete Control Model
3.4 MATRIX\textsubscript{x} Command File

The MATRIX\textsubscript{x} control simulation software allows the use of both an executable command text file and an online command structure format. Each line in the executable command file is accompanied by a corresponding line of comment text describing the commands function. The comment lines are denoted with double backslash lines at the left. Both the time step interval and the vehicle velocity are set at a factor of one, allowing the simulation to step through at a one second interval sampling rate. The cutter desired depth (CDD) is set at plus 3 inches, i.e. 3 in. cutting depth. As stated previously the terrain profile are phase shifted relative to each other. Both the left and right sensors are synthesised using sine waves. From the command file the distance shifts are adjusted to compensate for the ultrasonic signal to critical point distances. Through comparison with experiments on the profiler the gain for each of the three leg pistons was found to be 0.65. During the simulation two vector inputs are combined to provide a single matrix input. The final section of the command file sets the parameters for graphical plotting of the control system response. A complete listing of the entire executable command file is given in Appendix A.

3.5 Simulation Results

The output results from the MATRIX\textsubscript{x} control simulation are shown in Fig. 3.9. In each plot two outputs are graphed. In the first plot both the
left terrain input (LFT) and the left front elevator motion (LFE) are shown. In the second plot the right terrain input (RFT) and the right front elevator motion (RFE) are shown. It should be noted that the signal start in these two plots are both delayed by 72 inches, thus accounting for the distance between the terrain sensor and the front track. In each strip, the solid line is representative of the terrain that the vehicle traverses. The dashed line represents the hydraulic elevator travel, by which the controller attempts to compensate for both terrain variations and desired cutting depth (CDD) input. The third plot is a representation of both the cutter terrain and the actual cutter location. In this plot the cutter signal starts with a 193 inches delay, accounting for the terrain sensor to cutter distance. Similar to that of the front two plots the bottom plot is the response at the rear track. In the bottom plot the rear terrain changes are shown as well as the fact that the rear elevator stays in a desired stationary condition. Verification of the simulation output data shows that the cleaned terrain is consistently 3 in. below the original terrain profile.
Figure 3.9: MATRIXx Control Simulation Results
Chapter 4

Hardware Design

The PR-500-FL Pavement Profiler production model is equipped with ultrasonic and contact terrain sensing devices. These devices are mounted midway between the front tracks and the cutter center. Due to their location and design specific configuration they were unusable for this project. A number of components were required in order to mount the various sensors necessary to this project.

4.1 Sensor Configurations

A number of sensors are needed to acquire enough environmental and vehicle positioning data to allow accurate vehicle terrain following and cutter depth control. The position of each of the three hydraulic legs relative to a reference point, the profile of the terrain that is being traversed by the vehicle, and the vehicle tilt are required input data. A variety of sensing methods were investigated for leg elevation, terrain profile, and vehicle tilt
sensing. The method chosen for each type of data collection is expanded upon in the following subsections.

Three different types of sensor signals will be available for real time input to the system. Four **ultrasonic** sensors are used to evaluate the terrain that the vehicle is transversing. Two of these sensors will be located 72 inches ahead of and 72 inches above each of the front tracks. Two other ultrasonic sensors will be located behind the front tracks at a height of 24 inches, and will act as cross referencing signals to avoid terrain profile data errors due to desert vegetation and debris located in the vehicle track paths. The ultrasonic terrain profiles are to be sampled at regular distance intervals, e.g. every inch and stored. Elevation data must be accessed according to the distance between the sensor locations and that portion of the vehicle where terrain data is required. Integrating the vehicle velocity resulting in the distance traveled which is then used to synchronize the vehicle track locations with the actual terrain profile.

The second type of sensor inputs will be provided by linear displacement transformers (LDT) attached to each of the three leg hydraulic systems. They will provide real time displacement data pertinent to the vehicle's leg elevations and will be mounted atop the hydraulic legs. The velocity of the vehicle will be determined using a single LDT sensor located inside of the vehicle control cabin.

The third sensor input will be provided by the radiation array network
to be arranged in front of the vehicle by the EG&G radiation analysis team. Until such time as the EG&G radiation analysis team provides their input, a fixed cutting depth will be used for experimental purposes. Fig. 4.1 locates the position of each of the mountable sensors upon the vehicle.

![Diagram of PR-500-FL Pavement Profiler](image)

**Figure 4.1: Sensor Locations on the PR-500-FL Pavement Profiler**

### 4.1.1 Ultrasonic Sensors

Many methods of detecting the profile of a terrain exist. Some of these methods are: direct contact sensing, laser range finding, optical triangulation, and ultrasonic sensing. Due to the environmental conditions of the test site terrain (i.e. presence of vegetation), the direct contact sensing approach was not selected. Laser range finding was also not used because
airborne particles would prevent true and accurate distance measurement, which for this application is essential. Optical triangulation generates large inaccuracies because of varying terrain reflectivity and therefore was also not selected. Ultrasonic sensing was selected for the following reasons: (1) by using two comparatory sensors, the discrepancies presented by vegetation can be eliminated. (2) the airborne particles do not interfere with the broadcast or returning ultrasonic signals. and (3) it is relatively inexpensive and more rugged than other options.

Ultrasonic sensors are a non-invasive measuring method, using electromagnetic signals which can propagate through gaseous mediums. In nature, various species of bats use ultrasonic sensing to measure distances to the ground, prey, or cave walls. In the oceans of the world, dolphins and porpoise use ultrasonic sensing to determine the range to the shore, other colleges, or schools of fish. An ultrasonic tone signal is emitted from a point, and the time elapsed between the transmission and the echoed signal rebounding from the object is measured. With the known speed of sound in air the distance to the object is determined accurately. The sound speed varies considerably with temperature. Sensors with built-in temperature compensation are commercially available. Currently ultrasonic sensors are used in a wide variety of applications ranging from medical scanning and imaging systems to home protection configurations. The frequencies used in ultrasonic sensors range from the middle of the very low frequency band (VLF ≈ 20 kHz) to the very
high frequency band ($\nu_{HF} \approx 1.000$ MHz). Bats for example have been known to emit sound pulses with frequencies up to 200 KHz. Ultrasonic sensors can be coupled with personal computers in a variety of ways. In a paper published by Duchesne, Fischer, and Gray [5] an ultrasonic sensor was coupled directly to the printer port of an IBM PC compatible computer. This direct coupling was accomplished by the modification of the system's programmable interval time, its internal operating program, and activation of software written in the "C" programming language. Although this method works and is computationally extremely efficient, it can only be implemented using a single sensor. For this project multiple ultrasonic sensors (4) are required. The method for computer interfacing these multiple sensors and more will be discussed in a subsequent chapter.

After an extensive equipment search, the DCU-7E-8 Ultrasonic Sensor manufactured by Lundahl Instruments Incorporated was selected for terrain data acquisition. The determining factors in our selection of the DCU-7E-8 Ultrasonic Sensor were: (a) low power requirements. (b) temperature compensation. (c) rugged construction. (d) data acquisition interfacing capabilities, and (e) rapid sampling rates. As all ultrasonic devices, the DCU-7E-8 Ultrasonic sensors can be affected by several factors such as the target surface, distance to the target, and the size and angular inclination of the target. By mounting the sensors perpendicular to and at a distance of 3 to 6 feet from the ground we are able to surmount two of these three factors. Four
DCU-7E-8 Ultrasonic Sensors are mounted upon the vehicle. Two sensors are mounted on the left side of the vehicle, before and after the front left track. The remaining two are mounted similarly on the right side, before and after the front right track. The specifications for the DCU-7E-8 Ultrasonic Sensor are listed below in Fig. 4.2.

<table>
<thead>
<tr>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Characteristics</strong></td>
</tr>
<tr>
<td>Max. Output Voltage</td>
</tr>
<tr>
<td>Output Impedence</td>
</tr>
<tr>
<td>Max. Capacitance</td>
</tr>
<tr>
<td>Max. Cable Length</td>
</tr>
<tr>
<td>Current Draw</td>
</tr>
<tr>
<td>Transmit Frequency</td>
</tr>
<tr>
<td>Sample Rate</td>
</tr>
<tr>
<td>Sample Average</td>
</tr>
<tr>
<td>Operating Cycle</td>
</tr>
<tr>
<td>Input Voltage</td>
</tr>
<tr>
<td>Settling Time</td>
</tr>
<tr>
<td>Operating Distance</td>
</tr>
<tr>
<td>Accuracy</td>
</tr>
</tbody>
</table>

Figure 4.2: Table of DCU-7e-8 Ultrasonic Sensor Specifications

4.1.2 LDT Sensors

A requirement of this project is the ability to accurately locate the elevation of the vehicle above ground (i.e. the cutter to terrain distance) by
measuring the location of the hydraulic elevator legs. Several methods and devices for this solution were investigated. Position transducers, inductive displacement transducers, angular encoders using a rack and gear setup, linear variable differential transformer were evaluated. The Temposonics™ II Linear Displacement Transformer model was selected. The selection was based on the sensors rugged hostile environment design, analog output interface capabilities, and high precisional position measurement. The specifications for the Temposonics™ II Linear Displacement Transformer are listed in Fig. 4.3.

<table>
<thead>
<tr>
<th>Specifications</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Voltage</td>
<td>12 to 15 VDC</td>
</tr>
<tr>
<td>Current Draw</td>
<td>nominal = 90 ma average</td>
</tr>
<tr>
<td>Displacement</td>
<td>Up to 25 feet</td>
</tr>
<tr>
<td>Dead Space</td>
<td>2.5 inches</td>
</tr>
<tr>
<td>Nonlinearity</td>
<td>&lt;0.05% of full stroke</td>
</tr>
<tr>
<td>Repeatability</td>
<td>&lt;0.001% of full stroke</td>
</tr>
<tr>
<td>Frequency Response</td>
<td>50-200 kHz</td>
</tr>
<tr>
<td>Temperature Coefficient</td>
<td>10 ppm/°F(18 ppm/ °C)</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>0 °F to 158°F(-20 °C to 66 °C)</td>
</tr>
<tr>
<td>Max. Output Voltage</td>
<td>-10 to +10 VDC</td>
</tr>
</tbody>
</table>

Figure 4.3: Table of Linear Displacement Transformer Specifications

The LDT uses a momentary interaction of two magnetic fields to determine the location of a moveable rare earth magnet relative to a metal core. A single pulse width signal (launching pulse) is transmitted down the
length of the transducer tube and interacts with the rare earth magnetic field causing a torsional strain pulse to be detected in the sensing coils in the head assembly. The rare earth magnet location is calculated using the elapsed time between the launching pulse and the strain pulse. The model selected is equipped with a analog personality module (APM) built into the head assembly. The APM is an electronic circuit which converts a 16-bit digital output to an analog output. This module facilitates the interfacing between the sensor and the computer process interface.

A sensor housing was constructed for the mounting of the LDT to the Pavement Profiler. Two similar but dimensionally different designs were needed. The two front LDT’s used on the Pavement Profiler are of a different size than that used in the rear. The front LDT’s are 36 inches long, while the rear LDT is only 30 inches in length. The reason the rear sensor is shorter is because the hydraulic elevator in the rear leg of the vehicle has a shorter travel distance. The LDT mounts from the top of the housing with a screw through a collar (not shown), with the tip placed into a depression in the housing bottom. Fig. 4.4 shows a drawing of the housing design.
Housing Assembly #1
(2 Assy Required)

Housing Assembly #2
(1 Assy Required)

### Parts List for LVDT Sensor Housing Assemblies

<table>
<thead>
<tr>
<th>Part No.</th>
<th>Part Name</th>
<th>Assy #1</th>
<th>Assy #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Soc. Hd. Cap Screw</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>7</td>
<td>Magnet Collar Assy</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Washer, Split</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Hex Bolt</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>LVDT Support Rear</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>LVDT Support Front</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Housing, Main</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>LVDT Mount</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 4.4: Linear Displacement Transformer Housing Design
4.1.3 Tilt Sensor

During operation, the vehicle must not tilt excessively, nor must the soil conveyor boom be moved too far sideways as the vehicle traverses uneven terrain. Thus an information sensor is needed to measure the vehicle’s X and Y axis tilt, and to provide information to the computer program. A dual axial tilt sensor (SSY0018), manufactured by Spectron Systems Technology Inc., was selected to measure the tilt of the vehicle in the two axial planes. The SSY0018 has a built in signal conditioner, compatible voltage requirements, and the requisite operating temperature range. The tilt sensor’s signal is converted to an angle by the controller. The limits for the angular tilt of the vehicle will be determined experimentally. If the vehicle and conveyor boom together exceed the limits for angular tilt, an alarm will be sounded by the computer.

4.2 Console Cabinet Layout

The computer and electronics are housed in a standard 22 inch by 42 inch rack mount cabinet. The rack mount cabinet is screw mounted to the right side wall of the vehicle control cabin. The screw mounting of the cabinet also provides the necessary ground for the electrical systems. Located in the upper most section of the cabinet is a rack mounted, pull out, computer keyboard. Attached to the rear of the cabinet is a cable strain relief harness used to route the sensor and power cables into the cabinet. Located upon the
inside left wall of the cabinet are electrical bus bar connections for ground, +15 volts, and -15 volts. Both the computer and circuit board support trays are mounted on rollers to provide easy access. Below the computer 2 inches of closed cell foam rubber is used for shock isolation purposes. The computer is then strapped down to the computer support tray. Two AC fans, providing 215 cfm each, are mounted to both inside side panels to supply adequate air circulation for cooling of the computer and electrical circuitry. The circuit board support tray contains an AC to DC power converter, a bank of control relays and the D-A A-D converter expansion board. These components are all solid state and require no special shock and vibration protection. Lighted power switches for the +24 VDC and +12 VDC inputs are located on the outer left side of the cabinet. A layout of the computer cabinet is shown in Fig. 4.5.
Figure 4.5: Computer and Electrical Cabinet Diagram
4.2.1 Computer and Data Acquisition

To control the cutting depth of the PR-500-FL Pavement Profiler, a standard portable 486 computer was selected. This computer is light, compact, portable, and rugged enough to withstand the vibrational effects experienced within the vehicle control cabin. Another reason for the selection of a portable computer was the ease of removal during programming and revision. A report on the system specifications for the computer is listed in Appendix B. The production video card and monitor have been replaced with a touch screen video card and detachable flat screen plasma monitor allowing touch screen operation. The plasma monitor is mounted on the right hand side of the vehicle controls console.

A Data Translation, Inc. DT2831 Series software configurable analog and digital I/O board was selected for process data interfacing with the portable 486 computer. The DT2831 provides 16 input channels with 12-bit resolution for quick and accurate data sampling. The DT2831 also has software programmable direct memory access data transfer modes, an Am9513A System Timing Controller, and 2 independent D/A converters with 12-bit resolution. The DT2831 board has a 50 pin process interface connector. User connections are made on an expansion board which is mounted on the circuit board support. The electrical connections to the sensors and actuator switches are made there. Using the converter's analog to digital converter, sensor output signals are converted to 12-bit integer data accessible by the
The DT2S31 data acquisition card is capable of providing eight digital output commands. Each of these digital commands is in the form of either a zero or +2.5 VDC. As the program executes an output command the selected channel activates a control relay. The definitions for each output channel is given in Fig. 4.7.
### Table: Digital Output Channel Definitions

<table>
<thead>
<tr>
<th>Hydraulic Actuator</th>
<th>Channel #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydraulic Piston 0</td>
<td>0</td>
</tr>
<tr>
<td>Hydraulic Piston 1</td>
<td>1</td>
</tr>
<tr>
<td>Hydraulic Piston 2</td>
<td>2</td>
</tr>
<tr>
<td>Hydraulic Piston 3</td>
<td>3</td>
</tr>
<tr>
<td>Hydraulic Piston 4</td>
<td>4</td>
</tr>
<tr>
<td>Hydraulic Piston 5</td>
<td>5</td>
</tr>
<tr>
<td>Hydraulic Piston Conveyor Boom (right/left rot.) Left</td>
<td>6</td>
</tr>
<tr>
<td>Hydraulic Piston Conveyor Boom (right/left rot.) Right</td>
<td>7</td>
</tr>
</tbody>
</table>

Figure 4.7: Digital Output Channel Definitions

#### 4.2.2 Power and Relay Configuration

The air conditioning engine for the profiler is equipped with a +12 volt generator. A DC to AC converter is mounted to the floor of the vehicle control cabin along side of the computer cabinet. This converter has an input of +12 VDC, provided through a power switch, from the air conditioner's +12 volt generator. From the DC to AC converter an AC to DC power supply produces both plus and minus 15 VDC which is used in powering the various sensors. Since there are 8 digital outputs from the data acquisition interface, eight control relays are used to control the hydraulic elevator legs and the booms lateral hydraulic piston. The +24 VDC batteries of the vehicle's diesel engine are connected, through a power switch, to the output side of each of the control relays. The control relays which activate the hydraulic elevator
legs have variable input voltage activation coils, are solid state, and are rated at a current handling capacity of 12 amps.
Chapter 5
Software Design

The computer program for the control of the PR-500-FL Pavement Profiler was designed by Glen Hein, a computer science undergrad student of UNLV. The object-oriented “C++” programming language was used because it facilitates system interrupt processes and is highly portable. A Borland “C++” Rev. 3.0 compiler was used to create the executable program. The compiler utilizes an Integrated Development Environment (IDE) to combine the ease of windows operations with programming in the “C++” language. To facilitate the graphical interfacing of the program with the display the Quinn-Curtis Real Time Graphics and Measurement/Control Tools software was used.

5.1 Program Organization

The control program for the pavement profiler is systematically organized and commented where applicable. Below is a general listing of the programs
operation processes and routines.

- Listing of include files for main and subroutine usage.
- Definitions of global type classes.
- Prototyping of program subroutines.
- Definitions of global constants
- Record definitions for analog signal sampling.
- Start of main program and local definitions.
  
  1. Call to the routine to initialize the A/D board.
  2. Call to the routine to initialize and construct the control screen display.
  4. Setup of mouse activated menu buttons in control display.
  5. Analog signal sampling loop.
  6. Conversion of sampled signals to proper units.
  7. Arrange and update data for control screen display.
  8. Check for mouse events on menu buttons and send control-command selection.
  9. Setup and initialization of data record output files.
10. Closure of data record output files.


12. Update current mouse event and mouse window location.

13. Setup button status to reflect control-command.

14. Close graphics, files and end of main program.

• Routine to maintain fixed cutting depth.

1. Copy current inputs for output to files.

2. Leg Elevator decision (Up or Down).

3. Output data to a file.

4. Output digital signal to control relays.

• Routine to calibrate system to initial conditions.

1. Collect input signals.

2. Conversion of sampled signals to proper units.

3. Output data to a calibration file.

• Routine to initialize the A/D board.

1. Initialize the A/D board.


• Routine to setup graphics display.
1. Initialize graphics display parameters.

2. Setup mouse driver to report events.

3. Draw initial control screen display.

A complete listing of the program and the define file are listed in appendixes C and D. Upon execution the program displays a color representation of the black and white screen shown in Fig. 5.1.
Figure 5.1: Profiler Computer Control Screen
5.2 Computer Interface

The computer program interfaces with the real time controls of the PR-500-FL Pavement Profiler through the A/D converter and digital output channels of the data acquisition card. All sensors are connected using their respective installation procedures. Input from the sensors is converted to digital signals and read by the program. From the measurement input data the program determines the appropriate actuator position and updates its position with a digital command output. The digital output is routed to the control relays, which interface with existing actuator controls. The diagram in Fig. 5.2 shows the command and data routing for the profiler.

14 A/D analog input channels are sampled by the controller. Vehicle
forward velocity is input through channel 0. All other analog signal samples are input on separate channels according to the definitions shown in Fig. 4.6. Routine sampling occurs at regular distance intervals, e.g. every inch of profiler forward travel. A generalized control flow diagram outlining the typical command sequence for the control program is presented in Fig. 5.3.

### 14 A/D input Channels (0 through 13)

<table>
<thead>
<tr>
<th>Rotomill forward Vel.</th>
<th>INT(x dt)</th>
<th>Distance Traveled (in.)</th>
<th>Dist &gt; 1&quot; since last sample?</th>
<th>Read Channels 1 through 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Evaluation of Measured Data**

- If Tilt Sensor out of range —> Sound and Opt. Alarm
- Save meas. data in Ring Buffer and on Disk file
- Compute new output Reference data to the three hydraulic main Actuators (up/down)

For each hydraulic Actuator (i=1:4)
- If Ref(i) > Meas(i) THEN Set Upbit(i);
- If Ref(i) < Meas(i) THEN Set Downbit(i);

Create 8-bit digital Data word for Hydraulic Piston Control, and send to Digital Output.
- Only one bit can be set for any one valve. Switch to manual op. if interference by operator.

Figure 5.3: Profiler Computer Control Flowchart
Chapter 6

Experimental Results

A series of equations for the geometric model of the PR-500-FL Pavement Profiler were developed. From these equations a MATRIX\textsubscript{X} control simulation was constructed. This simulation proved that automatic control of the profiler was feasible. Conversion of the simulation to real time control of the profiler was accomplished using various sensors and computer hardware. A software package was written and installed on the profiler's portable computer. A series of both static and dynamic experiments were run using the computer program to control the cutter depth. The experiments were run at the area 6 location of the Nevada Test Site. This area of the test site is large open expanse with a relatively flat grade. The results of the static tests differed little from the dynamic ones, therefore only the dynamic tests are included in this thesis. The profiler was in motion for a distance of approximately 200 feet in each of the test runs. Directional control of the vehicle was performed by a licensed operator while the on-board computer
controlled the cutting depth. Initially the profiler’s hydraulic elevators were all set to a zero height condition according to manual height indicators in both the front and rear of the vehicle. After calibrating the program to these initial conditions, operation began and a desired cut depth was input. The top graphs in Figs. 6.1, 6.2, and 6.3 are representative of the changes in elevation for each of the three vehicle legs. The lower graphs in these same figures are the digital output commands issued to the hydraulic legs during the test runs.
Figure 6.1: 1 inch Cut Zero Start Depth
Figure 6.2: 2 inch Cut Zero Start Depth
Figure 6.3: 3 inch Cut Zero Start Depth
As seen in Figs. 6.1, 6.2, and 6.3, the profiler changed its hydraulic elevator position in order to implement the requested cutting depth. As the terrain sensed at the front ultrasonic sensors changed, the elevators proceeded to correct for these changes in order to maintain its fixed cutting depth.

The vehicle was then stopped and set to a level condition within the X-Y plane using an on-board leveling gauge built in to the profiler. The cutter was set at ground level by visual inspection and not by using the manual height indicators. The test was again implemented at a four inch cutting depth. The results are shown in Fig. 6.4. The profiler again performed as expected and removed the soil at the desired depth.
Figure 6.4: 4 inch Cutting Depth after Leveling
Chapter 7

Conclusions

7.1 Conclusions

A manually operated hydraulic pavement profiler was converted to automatic cutting depth control by installing a set of sensors and a digital control computer. All 9 sensors interface with the computer through an A/D interface. The installed sensor systems comprise: 4 ultrasonic terrain tracking sensors, 3 hydraulic elevator linear displacement transformers, 1 vehicle velocity linear displacement transformer, and 1 dual axis angular tilt sensor. The computer processes the measured data and controls the hydraulic actuators through eight digital output channels. The elevation of the three hydraulic actuators, and the conveyor boom rotation can be controlled from the computer. The controller maintains the cutting depth at a user-requested level. A graphical user interface was developed for interaction with the vehicle operator. Present user interfacing is through keyboard and mouse. A record of process data is automatically stored on the computer's hard disk.
hard disk. The system and all its components were installed and tested, and are functioning as designed. The cutting depth control algorithm was developed through an interactive simulation which included terrain elevation variations, the dynamics of the hydraulic actuators, and variable reference values of the cutting depth. The comparison of the simulation results with the actual experimental data show reasonably good agreement. Current differences between simulation and experiments are caused by time delays in the hydraulic system which are not yet included in the simulation model.

7.2 Recommendations

The following improvements and further R&D are recommended:

1. The design and testing of a terrain tracking algorithm, including high-pass filtering and vegetation (false signal) recognition, need development.

2. The development, installation, and integration of a radiation sensor array with both the vehicle and the digital computer is desirable.

3. The radioactive hazards to personnel could be reduced further by piloting the profiler vehicle remotely with RF radio control equipment.

4. In the event of acquisition of additional vehicles, a review of other manufacturer's products is suggested. For instance, the Gomaco 5000 series vehicles offer a number of potential benefits, such as cutter in
front of the vehicle (no soil compression before removal), rear conveyor boom - i.e. truck can follow the pavement trimmer on clean soil, rather than having to drive in front.
Bibliography


Appendix A

// Executable MATRIX$_{\text{x}}$ Control Program
// for PR-500 Pavement Profiler Simulation
dt=1
// Time Interval Length
VEL=1/dt;
// Terrain cycle period in inches (Vehicle Velocity)
t=[0:dt:2000*dt]';
// Time Steps
in=[3*ones(2001,1)];
// Cutter Desired Depth
TFL=72+sin(t/100);
// Front Left Terrain Sensed Input
TFR=72+sin(t/100);
// Front Right Terrain Sensed Input
FLD=72*(dt);
// Front Left Terrain Delay
FRD=72*(dt);
// Front Right Terrain Delay
CD=193*(dt);
// Cutter Terrain Delay
RD=288*(dt);
// Rear Terrain Delay
FLpist=.65
// Front Left Piston Gain
FRpist=.65
// Front Right Piston Gain
Rpist=.65
// Rear Piston Gain
u=[TFL,TFR,in];
// Multiple Input Merging
y=sim(t,u);
// Simulation Processing Parameters

// Simulation Results Plotting Section
plot('title/Profiler Cutter Control/ time date')
plot('xlabel/Sensor Travel in inches/'),
plot(y(:,9),y(:,1:8),'strip2 YLAB/FL(T&E) (in.)|FR(T&E)... (in.)|Cutter(T&O)(in.)|Rear(T&E) (in.)|/'),...
// Plot of Outputs
pause,erase
Appendix B

System Report
PR-500 Pavement Profiler Portable Computer Specifications

******************************
* System Summary *
******************************

------------------------- Computer -------------------------
Computer Name: IBM AT or compatible
Built-in BIOS: AMI, Sunday, April 4, 1993
Main Processor: Intel 80486, 66 MHz
Math Co-Processor: (Built-in)
Video Adapter: Video Graphics Array (VGA)
Mouse Type: Serial mouse, Version 8.00

------------------------- Disks -------------------------
Hard disks: 246M
Floppy disks: 1.44M, 1.2M

------------------------- Memory -------------------------
DOS Memory: 640K
Extended Memory: 15,360K
Expanded Memory: 1,024K

------------------------- Other Info -------------------------
Bus Type: ISA (PC/AT)
Serial Ports: 2
Parallel Ports: 1
Keyboard Type: 101-key
Operating System: DOS 6.00

******************************
* AUTOEXEC.BAT file from drive C: *
******************************

@ECHO OFF
PROMPT $p$g
PATH C:\;C:\DOS;C:\FASTBACK;C:\SPIDER;C:\BC\BIN;C:\WP51;C:\mouse.com
rem lh MOUSE CENHANCE BUS
rem LOGIMENU
SET TEMP = C:\DOS
dosedit
PROMPT $p$g
c:\mouse\mouse

******************************
* CONFIG.SYS file from drive C: *
******************************
BUFFERS=32,0
FILES=45
FCBS=4,0
BREAK=ON
LASTDRIVE=C
STACKS=0,0
SHELL=C:\COMMAND.COM /E:256 /p
device=c:\dos\himem.sys
DOS=HIGH,UMB
device=c:\dos\ansi.sys
device=c:\bc\dt\dt283x.sys
Appendix C

/* PR-500 Pavement Profiler
Automatic Control Program
Created by Glen Hein */

#include <conio.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <dos.h>
#include <time.h>
#include <graphics.h>
#include "define.h"
#include "rtgsubs.h"
#include "rtggraph.h"
#include "rtmeter.h"
#include "rttext.h"
#include "rtgcommo.h"
#include "rtmouse.h"
#include "qcprn.h"
#include "dtst_tls.h"
#include "dtst_tls.h"
#include "dtst_err.h"
#include "dtst_xmm.h"
#include "dtst_ctr.h"

#define ON 1
#define OFF 0
#define FCD 1
#define CALIB 2
#define TT 3
#define NONE 4

typedef struct _TData {
  SINGLE sing_data;
  float multiplier; /* Desired Unit/Voltage */
  float offset; /* Desired Unit */
  long raw_data; /* Raw data from board. NOT VOLTAGE.
  Must use Voltage() to convert. */
  float true_data; /* Desired Unit */
} TData;

typedef struct _TDataFile {

float main_data[14];
  unsigned char control_word;
} TDataFile;

void InitDisplay(rtstatpntr *rtstat);
void InitDataTran(void);
void CalibrateSystem(void);
void FixedCutDepth(void);

extern unsigned _stklen = 8192U; /* sets the stack size. */

rtstatpntr rtstat[32];
realtype _yy[16];
int linecolor[16],
  linefill[16];
FILE *data_file;
messagetype message[16];
int button_status[16];
int handle;
float desired_cutting_depth;
realtype updatedata[16];

TData data_table[] = {
  {{FORWARD_VELOCITY,FV_GAIN,FV_DAC},
   FV_MULT,FV_OFFSET,0,0},
  {{HYDR_FRONT_RIGHT,HFR_GAIN,HFR_DAC},
   HFR_MULT,HFR_OFFSET,0,0},
  {{HYDR_FRONT_LEFT,HFL_GAIN,HFL_DAC},
   HFL_MULT,HFL_OFFSET,0,0},
  {{HYDR_REAR_CENTER,HRC_GAIN,HRC_DAC},
   HRC_MULT,HRC_OFFSET,0,0},
  {{ANGLE_BOOM_LR,HBR_GAIN,HBR_DAC},
   HBR_MULT,HBR_OFFSET,0,0},
  {{ANGLE_BOOM_UD,HBU.GAIN,HBU.DAC},
   HBU_MULT,HBU_OFFSET,0,0},
  {{ULTRA_FRONT_RIGHT,UFR_GAIN,UFR_DAC},
   UFR_MULT,UFR_OFFSET,0,0},
  {{ULTRA_FRONT_LEFT,UFL_GAIN,UFL_DAC},
   UFL_MULT,UFL_OFFSET,0,0},
  {{ULTRA_CENTER_RIGHT,UCR_GAIN,UCR_DAC},
   UCR_MULT,UCR_OFFSET,0,0},
  {{ULTRA_CENTER_LEFT,UCL_GAIN,UCL_DAC},
   UCL_MULT,UCL_OFFSET,0,0},
  {{TILT_ROLL,TR_GAIN,TR_DAC},
   TR_MULT},

void main()
{
    realtype avalue;
    int mouse_x,
        mouse_y,
        mouse_status,
        mouse_count,
        result,
        channelnumber,
        windownumber,
        i,
        control_command,
        loop_var;
    DIGITALIO digitalio;
    char buffer[40];
    time_t secs;
    struct tm *date;
    int record_signal;

    InitDataTran();
    InitDisplay(rtstat);

    digitalio.port = 1;
    digitalio.command = 0;
    digitalio.direction = 1;
    digitalio.dio_value = NO_OP;
    dt_set_dio(BOARD_NUMBER,
              handle,
              &digitalio);
    buffer[0] = 0;

    *** Mouse Command and Activation Section ***

    for(i = 0; i < 16; i++)
{
  button_status[i] = OFF;
  linecolor[i] = 1;
  linefill[i] = 0;
}
strcpy(message[0], "Quit RotoMill Control System");
sprintf(message[1], "Record RotoMill Data - %s", buffer);
strcpy(message[2], "Record RotoMill Control System Data");
strcpy(message[3], "Calibrate RotoMill System");
strcpy(message[4], "Maintain Fixed Cut Depth");
strcpy(message[5], "Maintain Terrain Tracking");
value = 0;
desired_cutting_depth = value;
record_signal = OFF;
while(button_status[0] == OFF)
{
    /* Read all analog inputs */
    for(loop_var=F0RWARD_VEL0CITY;loop_var<=ULTRA_BOOM_LEFT;
        loop_var++)
    {
        dt_single_acq(BOARD_NUMBER,
                      handle,
                      AD_SECTION,
                      &(data_table[loop_var].raw_data),
                      &(data_table[loop_var].sing_data));

        data_table[loop_var].true_data =
Voltage(data_table[loop_var].raw_data)
* data_table[loop_var].multiplier
+ data_table[loop_var].offset;
    }

    updatedata[0] = data_table[HYDR_FRONT_LEFT].true_data;
    updatedata[1] = data_table[HYDR_FRONT_RIGHT].true_data;
    rtupdatedisplay(rtstat[0], updatedata);
    updatedata[0] = data_table[FORWARD VELOCITY].true_data;
    rtupdatedisplay(rtstat[1], updatedata);
    updatedata[0] = data_table[ANGLE BOOM_LR].true_data;
    rtupdatedisplay(rtstat[2], updatedata);
    updatedata[0] = data_table[TILT ROLL].true_data;
    rtupdatedisplay(rtstat[3], updatedata);
    /* CASE out for current algorithm.
1. Terrain Tracking
2. Fixed Cutting Depth
3. Calibrate


4. No action

    */
    control_command = NONE;
    if(button_status[2] == ON)
        control_command = CALIB;
    if(button_status[3] == ON)
        control_command = FCD;
    if(button_status[4] == ON)
        control_command = TT;
    if(record_signal == ON)
    {
        if(button_status[1] == ON)
        {
            time(&secs);
            date = localtime(&secs);
            sprintf(buffer, "yo_dyo_d_RSD", date->tm_hour,
            date->tm_min,date->tm_sec);
            data_file = fopen(buffer, "wb");
            record_signal = OFF;
        }
        if(button_status[1] == OFF)
        {
            fclose(data_file);
            record_signal = OFF;
        }
    }
    switch(control_command) {
    case TT:
        break;
    case FCD:
        FixedCutDepth();
        break;
    case CALIB:
        CalibrateSystem();
        break;
    case NONE:
        break;
    }

    rtgetmouse(&mouse_status, &mouse_x, &mouse_y);

    channelnumber = rtgetwindowchannel(rtstat, 5,
    5,
mouse_x,
mouse_y,
&windownumber);
    if(mouse_status == 1)
{
    result=rtfindmousewin(rtstat,0,4,mouse_x,mouse_y);
    if(result == 4)
    {
        rtgetsliderstate(rtstat[4],mouse_x,mouse_y, &avalue);
    }
    rtmousepressed(0,&mouse_status,&mouse_count, &mouse_x, &mouse_y);
    if(channelnumber != -1 && mouse_count == 1 )
    {
        if(button_status[channelnumber] == ON)
    button_status[channelnumber] = OFF;
    else
    button_status[channelnumber] = ON;
    if(channelnumber == 3)
    {
        if(button_status[3] == ON)
    button_status[4] = OFF;
    }
    if(channelnumber == 4)
    {
        if(button_status[4] == ON)
    button_status[3] = OFF;
    }
    if(channelnumber == 2)
    {
    button_status[3] = OFF;
    button_status[4] = OFF;
    }
    if(channelnumber == 1)
    record_signal = ON;
    for(i = 0; i < 5; i++)
    linefill[i] = 3 * button_status[i];
    rthidemouse();
    rtupdategttext(rtstat[5],message,linecolor, linefill);
    rtshowmouse();
}
void FixedCutDepth()
{
    int i;
    TDataFile file_data;
    unsigned char output_data;
    DIGITALIO digitalio;

    for(i = FORWARD VELOCITY; i < ULTRA_BOOM LEFT; i++)
    {
        file_data.main_data[i] = data_table[i].true_data;

        output_data = NO_OP;
        if((data_table[HYDR_FRONT_RIGHT].true_data +
            desired_cutting_depth) > MAX_CUT_ERROR)
            output_data |= HYDR_FR_DN;
        else
            if((data_table[HYDR_FRONT_RIGHT].true_data +
                desired_cutting_depth) < MAX_CUT_ERROR)
                output_data |= HYDR_FR_UP;

        if((data_table[HYDR_FRONT_LEFT].true_data +
            desired_cutting_depth) > MAX_CUT_ERROR)
            output_data |= HYDR_FL_DN;
        else
            if((data_table[HYDR_FRONT_LEFT].true_data +
                desired_cutting_depth) < MAX_CUT_ERROR)
                output_data |= HYDR_FL_UP;

        if((data_table[HYDR_REAR_CENTER].true_data +
            desired_cutting_depth) > MAX_CUT_ERROR)
            output_data |= HYDR_RC_DN;
        else
            if((data_table[HYDR_REAR_CENTER].true_data +
                desired_cutting_depth) < MAX_CUT_ERROR)
                output_data |= HYDR_RC_UP;

    }
file_data.control_word = output_data;
fwrite(&file_data, sizeof(TDataFile), 1, data_file);
digitalio.dio_value = output_data;
dt_set_dio(BOARD_NUMBER,
    handle,
    &digitalio);
}

*** Calibration of System ***

void CalibrateSystem()
{
    int loop_var;
    FILE *fp;

    fp = fopen("CALIBRAT.DAT", "wt");
    for(loop_var=FORWARD_VELOCITY;loop_var<=ULTRA_BOOM_LEFT;
        loop_var++)
    {
        data_table[loop_var].offset=
            (Vtage(data_table[loop_var].raw_data) *
            data_table[loop_var].multiplier);
        if(fp)
            fprintf(fp,
                "Channel #%d:\tOffset = %f\n",
                loop_var,
                data_table[loop_var].offset);
    }
    fclose(fp);
}

*** Initialization of Data Transfer Function ***

void InitDataTran()
{
    int errstatus;
    BOARDS board;
    char *device_name = "DT283X$0";

    board.dma_channel_1 = DMA_CHANNEL_1;
    board.dma_channel_2 = DMA_CHANNEL_2;
    board.interrupt_level = INTERRUPT_LEVEL;
    board.board_timeout = TIMEOUT;
board.ad_setup_bits = AD_INPUTS;
/* Set single-ended inputs */
board.da_setup_bits = 0;
/* No D/A setup */
errstatus = dt_initialize(device_name, &handle);
if(errstatus == errComplete) {
    errstatus = dt_reset(BOARD_NUMBER, handle);
    errstatus = dt_set_board(BOARD_NUMBER, handle, &board);
}

*** Start of Graphic Interface Function ***

void InitDisplay(rtstatpntr *rtstat)
{
    tagtype tags[16],
alarmt[16],
halarmt[16],
nalarmt[16],
units[16],
title[16];
    int linecolor[16],
linefill[16],
xdecimals,
ydecimals,
numbertraces,
grid,
startangle,
stopangle,
farray[16],
status,
buttons,
batchflag,
segments,
nhtic,
numberrows,
nnumbercolumns,
fontsize;
    realtype timeinterval,
sampleinterval,
miny,
maxy,
resetinterval,
cpx,
cpy,
radius,
lalarm[16],
halarm[16],
setpoint[16],
startvalue[16],
min[16],
max[16],
increment[16];
  alarmstrtype  alarmstrings[3];
  messagetype   message[16];

rtinitgraphics("C:\BC\BGI", 7, rtstat, 1);

rtsetrealbufferflag(rtstat[0], 1);
rtinitmouse(&status, &buttons);
rtshowmouse();

rtsetpercentwindow(rtstat[0], 0.00, 0.00, 1.00, 0.33);
rtsetpercentwindow(rtstat[1], 0.00, 0.33, 0.33, 0.66);
rtsetpercentwindow(rtstat[2], 0.33, 0.33, 0.66, 0.66);
rtsetpercentwindow(rtstat[3], 0.66, 0.33, 1.00, 0.66);
rtsetpercentwindow(rtstat[4], 0.00, 0.66, 0.33, 1.00);
rtsetpercentwindow(rtstat[5], 0.33, 0.66, 1.00, 1.00);

xdecimals = 0;
ydecimals = 0;
timeinterval = 10.0;
sampleinterval = 1.0 / 11.0;
miny = -20.0;
maxy = 20.0;
resetinterval = 0.95;
numbertraces = 2;
grid = 0;
lalarm[0] = -22.0;
halarm[0] = 22.0;
setpoint[0] = 0.0;
linecolor[0] = 1;
linecolor[1] = 4;
linefill[0] = 0;
linefill[1] = 0;
strcpy(tags[0], "Left LVDT");
strcpy(tags[1], "Right LVDT");
strcpy(units[0], "Inches");
strcpy(title[0], "Front LVDT's Display");
ratchflag = 0;
rtsetupsweepgraph(rtstat[0],
timeinterval,
sampleinterval,
miny,
maxy,
resetinterval,
numbertraces,
grid,
lalarm[0],
halarm[0],
setpoint[0],
xdecimals,
ydecimals,
title[0],
units[0],
tags,
linecolor,
linefill,
ratchflag);

linecolor[0] = 1;
linefill[0] = 3;
strcpy(tags[0], "" );
strcpy(alarmstrings[0], "Speed Too Low");
strcpy(alarmstrings[1], "Speed Normal");
strcpy(alarmstrings[2], "Speed Too High");
strcpy(title[0], "Vehicle Velocity");
strcpy(units[0], "feet/min");
miny = 0;
maxy = 25;
lalarm[0] = 2.5;
halarm[0] = 22.5;
setpoint[0] = 20;
cpx = 0.5;
cpy = 0.5;
radius = 0.3;
startangle = 0;
stopangle = 180;
xdecimals = 0;
numbertraces = 1;
segments = 5;
nthtic = 1;
rtsetupmeter(rtstat[1],
  miny,
  maxy,
  cpx,
  cpy,
  radius,
  startangle,
  stopangle,
  lalarm[0],
  halarm[0],
  setpoint[0],
  xdecimals,
  title[0],
  units[0],
  tags,
  alarmstrings,
  numbertraces,
  segments,
  nthtic,
  linecolor,
  linefill);

    linecolor[0] = 1;
    linefill[0] = 3;
    strcpy(tags[0], "")
    strcpy(alarmstrings[0], "")
    strcpy(alarmstrings[1], "")
    strcpy(alarmstrings[2], "")
    strcpy(title[0], "Boom Angle")
    strcpy(units[0], "degrees")
    miny = -60;
    maxy = 60;

    lalarm[0] = -60;
    halarm[0] = 60;
    setpoint[0] = 0;
    cpx = 0.5;
    cpy = 0.5;
    radius = 0.3;
    startangle = 45;
    stopangle = 135;
    xdecimals = 0;
    numbertraces = 1;
segments = 4;
nthtic = 1;
rtsetupmeter(rtstat[2],
    miny,
    maxy,
    cpx,
    cpy,
    radius,
    startangle,
    stopangle,
    lalarm[0],
    halarm[0],
    setpoint[0],
    xdecimals,
    title[0],
    units[0],
    tags,
    alarmstrings,
    numbertraces,
    segments,
    nthtic,
    linecolor,
    linefill);

linecolor[0] = 1;
linefill[0] = 0;
strcpy(units[0], "degrees");
strcpy(lalarm[0], "Stable");
strcpy(halarm[0], "DANGER");
strcpy(nalarm[0], "Stable");
lalarm[0] = -1;
halarm[0] = 30;
setpoint[0] = -1;
strcpy(title[0], "Tilt Detection");
farray[0] = 1;
farray[1] = 1;
farray[2] = 1;
farray[3] = 1;
strcpy(tags[0], "Latitude");
xdecimals = 0;
numberrows = 1;
numbercolumns = 1;
rtsetupannun(rtstat[3],
lalarm,
halarm,
setpoint,
xdecimals,
title[0],
units,
tags,
halarmt,
lalarmt,
nalarmt,
numberrows,
numbercolumns,
linecolor,
array);

xdecimals = 1;
strcpy(title[0], """);
strcpy(tags[0], "Cut Depth");
strcpy(units[0], "Inches");
linecolor[0] = 1;
startvalue[0] = 0;
min[0] = -12;
max[0] = 12;
increment[0] = 0.1;
fontsize = 1;
rtsetupslider(rtstat[4],
1,
title[0],
units,
tags,
1,
1,
startvalue,
min,
max,
increment,
fontsize,
linecolor);

strcpy(title[0], "RotoMill System Options");
strcpy(message[0], "Quit RotoMill Control System");
strcpy(message[1], "Record RotoMill Control System Data");
strcpy(message[2], "Calibrate RotoMill System");
strcpy(message[3], "Maintain Fixed Cut Depth");
strcpy(message[4], "Maintain Terrain Tracking");
numberrows = 5;
numbercolumns = 1;
linecolor[0] = 1;
linecolor[1] = 1;
linecolor[2] = 1;
linecolor[3] = 1;
linecolor[4] = 1;
linefill[0] = 0;
linefill[1] = 0;
linefill[2] = 0;
linefill[3] = 0;
linefill[4] = 0;
rtsetupgtext(rtstat[5],
title[0],
message,
numberrows,
numbercolumns,
linecolor,
linefill);
Appendix D

/* PR-500 Pavement Profiler
   Automatic Control Definition Program
   Created by Glen Hein */

/* ------------------------ Define List ------------------------*/
/* palette memmory address */
#define PEL_ADDR 0x03C8
#define PEL_DATA 0x03C9

/* Pathnames to external data files */
#define OPENSCREEN "OPEN.SCR"
#define BOOMYN "BOOMYN.SCR"
#define BOOMM "BOOMM.SCR"
#define PALFILE "ROTOMILL.PAL"
#define FONTFILE "MAIN.FNT"

/* Delay for openning screen */

/* Data for all sensors:
The first value is the channel number to read.
The second value is the gain that should be applied to the Data Translation Board.
The third value is the DAC to be used.
No DACS will be used.
However, all need to be set to zero.
The fourth value is the value to be multiplied with the voltage.
The fifth value is the value to be added to the product of the multiplier and voltage. */

#define FORWARD_VELOCITY 0
#define FV_GAIN 0
#define FV_DAC 0
#define FV_MULT 0 /*mph/Volts*/
#define FV_OFFSET 0 /*mph*/
#define HYDR_FRONT_RIGHT 1
#define HFR_GAIN 0
#define HFR_DAC 0
#define HFR_MULT -3.6 /*Inches/Volts*/
#define HFR_OFFSET 18 /*Inches*/
/* A negative value for the true_data on any LDT signifies that the vehicle is "lower" than ground level. */

#define HYDR_FRONT_LEFT 2
#define HFL_GAIN 0
#define HFL_DAC 0
#define HFL_MULT -3.6 /*Inches/Volts*/
#define HFL_OFFSET 18 /*Inches*/

#define HYDR_REAR_CENTER 3
#define HRC_GAIN 0
#define HRC_DAC 0
#define HRC_MULT 0 /*Inches/Volts*/
#define HRC_OFFSET 18 /*Inches*/

#define ANGLE_BOOM_LR 4
#define HBR_GAIN 0
#define HBR_DAC 0
#define HBR_MULT 0
#define HBR_OFFSET 0

#define ANGLE_BOOM_UD 5
#define HBU_GAIN 0
#define HBU_DAC 0
#define HBU_MULT 0
#define HBU_OFFSET 0

#define ULTRA_FRONT_RIGHT 6
#define UFR_GAIN 0
#define UFR_DAC 0
#define UFR_MULT 19.2 /*Inches/Volts*/
#define UFR_OFFSET 19.2 /*Inches above ground*/

#define ULTRA_FRONT_LEFT 7
#define UFL_GAIN 0
#define UFL_DAC 0
#define UFL_MULT 19.2 /*Inches/Volts*/
#define UFL_OFFSET 19.2 /*Inches*/

#define ULTRA_CENTER_RIGHT 8
#define UCR_GAIN 0
#define UCR_DAC 0
#define UCR_MULT 0 /*Inches/Volts*/
#define UCR_OFFSET 0 /*Inches*/

#define ULTRA_CENTER_LEFT 9
#define UCL_GAIN 0
#define UCL_DAC 0
#define UCL_MULT 0 /*Inches/Volts*/
#define UCL_OFFSET 0 /*Inches*/

#define TILT_ROLL 10
#define TR_GAIN 0
#define TR_DAC 0
#define TR_MULT 0
#define TR_OFFSET 0

#define TILT_PITCH 11
#define TP_GAIN 0
#define TP_DAC 0
#define TP_MULT 0
#define TP_OFFSET 0

#define ULTRA_BOOM_RIGHT 12
#define UBR_GAIN 0
#define UBR_DAC 0
#define UBR_MULT 0
#define UBR_OFFSET 0

#define ULTRA_BOOM_LEFT 13
#define UBL_GAIN 0
#define UBL_DAC 0
#define UBL_MULT 0
#define UBL_OFFSET 0

/* Output Channels */
#define NO_OP 0x00
#define HYDR_FR_UP 0x01 /* Bit 0 */
#define HYDR_FR_DN 0x02 /* Bit 1 */
#define HYDR_FL_UP 0x04 /* Bit 2 */
#define HYDR_FL_DN 0x08 /* Bit 3 */
#define HYDR_RC_UP 0x10 /* Bit 4 */
#define HYDR_RC_DN 0x20 /* Bit 5 */
#define CB_RL_RT 0x40 /* Bit 6 */
#define CB_RL_LT 0x80 /* Bit 7 */

#define BOARD_NUMBER 0
```c
#define TRUE 1
#define FALSE 0

/* The width and height in pixels of the current display type. SCREEN_BUFFER_SIZE should be width*height. */
#define SCREEN_BUFFER_SIZE 64000
#define SCREEN_WIDTH 320
#define SCREEN_HEIGHT 200

/* This is the number of Data Translation Boards installed in the system. */
#define NUMBER_OF_BOARDS 1
#define DMA_CHANNEL_1 0
#define DMA_CHANNEL_2 0
#define INTERRUPT_LEVEL 10
#define TIMEOUT 0

/* Increase this value to make letters farther apart. */
#define FONTSPACING 1

/* Predefined color chart. */
#define BLACK 0x00
#define WHITE 0xFF
#define OTHER 0x25

/* The number of inches to move before another systems wide evaluation occurs. */
#define CHECK_DISTANCE 1

/* This value should be tested for each new computer system. It is the number of milliseconds the main rountines in the MainAppLoop requires. */
#define TIMER_INTERRUPT 8
#define PIT_FREQ 1193181
```
/* This is the number of CHECK_DISTANCE units that should equal the wheel base of the rotomill plus distance to front ultrasonic sensor. */
#define RING_BUFFER_SIZE 288

/* Number of inches to cut. Negative values imply cutting. */
#define DEFAULT_CUTTING_DEPTH -4

/* This is a flag value used in the program. */
#define DONT_CHANGE_DEPTH 999

/* A predefined macro for converting raw data from the Data Translation Board into a true voltage. */
#define UPPER_X 10
#define UPPER_Y 10

#define NORMAL_RUN 0x01
#define IDLE 0x02
#define CALIBRATE 0x04
#define FIXED_CUT_DEPTH 0x08
#define EXEC_MAIN_LOOP 0x10
#define SET_CUT_DEPTH 0x20

/* Min speed for normal operation. */
#define MIN_SPEED 1
#define MAX_CUT_ERROR 1

#define REAR_TERRAIN_DIST 288

#define Voltage(data) (((float)(data))*5/1024)