

DEVELOPING AN INTERACTIVE METHOD OF DISPLAYING PAVEMENT MANAGEMENT INFORMATION USING SAS/AF AND PROC GMAP

Bruce Dietrich, Florida Department of Transportation

ABSTRACT

Highway engineers must analyze large volumes of roadway data as part of the process of maintaining our highway system. An interactive Pavement Management Reporting System has been developed at the Florida Department of Transportation using SAS/AF and PROC GMAP to facilitate this process. Techniques were developed to dynamically access and process the data and to display the information as color coded geographic roadway maps. The system uses GMAP features such as unit areas, response data sets, annotations, legends and titles to generate the customized maps based on user selections. The application uses numerous SAS/AF program entries to allow user selected branching and to facilitate modular program development and modification. Pushbutton fields and pop up selection lists are used to create graphical user interfaces so the system can easily be used by non-programmers.

INTRODUCTION

The Florida Department of Transportation (FDOT) has approximately 37,000 lane miles of roadway that it is responsible for maintaining. Part of this responsibility includes keeping the pavement surfaces in a good and serviceable condition for the traveling public. Over time, these pavements will wear out due to the effects of vehicle loads and the environment on pavement materials, requiring the roads to be resurfaced. To schedule resurfacing projects so that the most effective use is made of the funding available, the FDOT collects an extensive amount of data. This includes annual condition surveys on each section of road, extensive traffic data collection of vehicle counts, types, and weights and roadway inventory information, such as pavement widths, shoulders, etc. This information is utilized by engineers in the FDOT's Districts and

Central Office to develop, and keep updated, a five year work program that is maintained in a mainframe database.

To assist the engineers in this process, the FDOT has developed a PAVement Management Reporting System (called PAVMARS) using the SAS Software System. This system uses SAS's data access capabilities to extract pavement information from various database locations. The data is then combined and summarized using various programs written in SAS software, and then presented in the form of color coded geographic maps, as well as reports and charts.

Since many non-programmers use the PAVMARS system, SAS/AF has been used extensively to create graphical user interfaces that generally only require the user to move the cursor and select a pushbutton option on the computer screen. This paper will briefly describe some of the techniques that were used in developing the PAVMARS system.

NEED FOR A PAVEMENT MANAGEMENT REPORTING SYSTEM

The FDOT has been in the process of gradually automating its operational functions for many years. An extensive set of databases and software systems have evolved around each of the functional areas of the Department as they have become automated. However, since the process has been gradual and technology has changed rapidly, these systems were not all designed to talk to each other, and the output and user interfaces have mostly been character based. This usually is adequate to serve the needs of a specific functional area such as Construction or Management and Budget, since a set of reports were developed along with system, tailored to the needs of that office. This is not adequate for a pavement management system, which must rely on data generated by a variety of offices. With the tremendous volume of data

available and that must be analyzed for pavement management, additional techniques for summarizing, analyzing and presenting the data are needed. With the rapid evolution of technology, new sources of pavement data are constantly being added, and the pavement management system must be flexible for quick modification and expansion. The tools available with the SAS System have proven very effective in accomplishing this difficult task.

DATA PROCESSING AND ORGANIZATIONAL ENVIRONMENTS AT FDOT

The FDOT has a large IBM mainframe computer running the MVS operating system in the central Tallahassee office. The mainframe is networked throughout the state to the eight FDOT District Offices and numerous Construction, Maintenance and other satellite offices. Both terminals and PCs with terminal emulation capabilities are used to access the mainframe.

The FDOT is a decentralized organization, with the District Offices responsible for carrying out their respective work programs, and the Central Office responsible for developing uniform procedures, and conducting quality assurance reviews and training. Each District and the Central Office is also organized into functional areas such as Design, Planning, Construction, Materials, etc.

Most data processing needs involving mainframe database updates are programmed by the central Office of Information Systems at the request of the various functional areas. However, as the need for data access and analysis has grown, many of the report writing needs are being met within the functional area offices through the use of end-user programming, primarily using the SAS software system.

Pavement Management is a process that crosses many of the functional lines within the Department. Data input and data access regarding pavement management data involves many offices and locations. These needs are coordinated by a statewide Pavement Management Policy Committee that meets periodically to set policy and assign tasks and responsibilities. The FDOT Pavement Management Office provides staff sup-

port to this committee, as well as providing Pavement Design functional responsibilities.

PAVMARS SYSTEM DESIGN

To meet many of the data analysis and reporting needs of pavement management, the PAVement Management Reporting System (PAVMARS) has been developed by staff within the Central Pavement Management Office.

The system accesses pavement data from a variety of data sources. These include many different mainframe DB2 database tables and views, as well as SAS data sets and flat files. The system is written entirely in SAS and is maintained by the Central Pavement Management Office in Tallahassee. Access to the system is available throughout the FDOT statewide computer network.

Since the system is accessed by many different users, with varying levels of computer expertise, graphical user interfaces are used exclusively. SAS/AF program entries with pushbuttons and pop-up selection lists are used to create this environment and to minimize the chance for user input error.

Screen control language (SCL) is used within the program entries to process the user selections and provide branching between program entries. PAVMARS was originally written in SAS version 5 and converted to version 6. Since SCL was not available in version 5, base SAS code was used in the program entries and is still being used within submit blocks in the current version. However, these submit blocks are gradually being converted to SCL code where possible. Significant performance improvements have been noted by using compiled SCL code over submit blocks, and the debugging of code is easier using the SCL debugger and the TESTAF command.

A modular system design was used for PAVMARS with various program functions placed in different AF program entries. This modular design initially aided in developing and debugging the original version of the system. The maintenance and expansion of the system is also greatly facilitated through the use of program entry modules. The code in any

individual program entry is kept to no more than a few hundred lines so that its logic can be followed easily. When making modifications or enhancements to the system, backup copies of the entries affected can be made quickly and problems isolated by using the SCL interactive debugger.

Like most data processing systems, PAVMARS modules can be broken down into three general functions. These are Input, Processing and Output. Since continual expansion and enhancement of the system was anticipated, the processing modules were written generically to handle a variety of pavement features. The input and output functions were divided into generic and customized modules.

Typically, to add a new feature to the PAVMARS system, a single feature selection module is added to the input portion, and a single variable selection module is added to the output portion.

These modules are processed in the following order.

- o Generic area selection
- o Customized feature selection
- o Generic map generation
- o Customized feature variable selection
- o Generic map display
- o Generic output and replay options

This is accomplished by extensive use of macros and macro variables and a generic feature response data set structure that is passed between the various modules. Based on the feature and area selected by the user, data is extracted and processed from the appropriate databases for that feature. The complexity of the data extraction and processing varies greatly between different features, depending on the number of databases being accessed and the type of analysis being done.

Roadway information within the FDOT is referenced to the same roadway identification and milepost numbering system for all databases. This allows a

generic feature response data set to be built by each feature selection module, using the same location referencing system. The response data set can contain up to six different variables describing each individual section of roadway for a particular feature. Macro variables describing the area and feature are assigned in the feature selection modules.

In the customized feature variable selection module, additional macro variables are assigned, based on user selection, to control titles, legends, data ranges and colors in the generic map display module.

DYNAMIC SEGMENTATION

PROC GMAP is used by PAVMARS to display the roadway feature information as color coded geographic maps. PROC GMAP uses three data sets to generate a map. These are a map data set, a response data set, and an annotation data set. The map data set contains unit areas defined by a series of coordinate observations with identifying key variables. The response data set contains descriptive data values pertaining to the unit areas and key variables to link the data values to the map unit areas. The annotation data set contains coordinates and graphic instructions for adding labels and other information to the map.

As mentioned previously, the FDOT uses a roadway identification number (rdwyid) and milepost referencing system to locate roadway information. A digitized map has also been created for all the state maintained roads. This map is stored as a SAS data set with a roadway id, milepost and x,y coordinates for each digitized point. There are approximately 90,000 points in the state maintained roadway map. Mileposts are stored to three decimal places.

The digitized point locations were placed manually using computer-aided design software and vary with the curvature of the road. These digitized points usually do not correspond to the milepost limits of feature information such as pavement condition or project locations that are desired to be displayed.

To display the feature information at

its true boundaries, a process called dynamic segmentation is used. This process involves locating a feature observation, then searching through the map database to find digitized points on either side of the feature limits. When these points are found, a new point is created at the boundary by interpolating between the digitized points. This process is carried out dynamically after the user makes a feature selection, and breaks the digitized map up into segments corresponding to the feature limits; thus the term dynamic segmentation.

Since the user is waiting for this process to finish, it is important that it be accomplished quickly. A companion data set to the digitized map file is used to speed the access process. This data set contains the first and last observation numbers for each roadway id in the map file. It is merged in with the feature response data set and then the SET statement POINT= option is used to randomly access the map data set as each roadway id in the response data set is processed. The following partial code shows how the dynamic segmentation is carried out.

```
/* featresp is response data */
/* set and contains rdwyid, */
/* beg, end and response data */
/* for each roadway in the */
/* selected area. beg and */
/* end are milepost limits. */

/* map.random contains rdwyid, */
/* fob,lob (1st and last obs */
/* for each rdwyid in map) */

/* map.roadnet is the map and */
/* contains rdwyid, mp, x, y */
/* (milepost and coords) */

data featresp;
merge featresp map.random;
by rdwyid;

/* featnet is new data set */
/* of map coords for each */
/* feature response location */

data featnet;
set featresp;

/* initialize previous pt */
/* coords for each response */
/* data set observation */
xmp=.
```

```
xx=.;
xy=.;

/* set response limit flags */
begfound=0;
endfound=0;

/* process 1st to last map pts */
/* by random access for each */
/* rdwyid obs in resp dataset */

do i=fob to lob;
set map.roadnet point=i;

/* hold current map pt info */
hx=x;
hy=y;
hmp=mp;

/* look for beg limit */
if not begfound then do;
/* if beg limit found */
if mp ge beg then do;
if mp=beg then output;
else do;
/* interpolate coords */
d=beg-xmp;
l=hmp-xmp;
ratio=d/l;
dx=hx-xx;
dy=hy-xy;
x=xx+dx*ratio;
y=xy+dy*ratio;
mp=beg;
/* output interpolated pt */
output;
/* restore held map pt */
x=hx;
y=hy;
mp=hmp;
end; /* mp ne beg */
begfound=1;
end; /* mp ge beg */
end; /* not begfound */

/* output intermediate pts */
output;

/* similar code for end pt */
/* would go here(not shown) */

/* store previous map pt */
/* for next iteration */
xmp=mp;
xx=x;
xy=y;
end; /* do i=fob to lob */
```

After creating the featnet data set containing the digitized roadway centerline of each feature data response location, some additional processing is

needed.

PROC GMAP uses unit area polygons for response map areas. Therefore, the linear roadway points must be converted into a polygon. This is accomplished by creating duplicate points in reverse order to the digitized points. During this process, offsets are added for right and left roadways and optionally to increase the displayed width of the roadways.

RESPONSE VARIABLE SELECTION

After the feature response map is created, an annotation data set is created. These data sets must be projected using PROC GPROJECT to convert the spherical coordinates (latitude and longitude) into cartesian coordinates. Once the map is projected, the user is shown a customized window for variable selection corresponding to the feature previously selected. Branching to these windows is controlled by a macro variable set when the user selected the feature.

Up to six different data values for each unit area can be contained in the feature response data set. The customized variable selection program sets macro variables for data ranges and colors to be displayed on the map, based on user selections. This allows several different maps to be generated by the user without having to re-extract data and rebuild the map and annotation data sets. An example would be pavement condition data that has several different types of distress measurements for each roadway section.

BRANCHING

After a map is displayed on the screen and terminated, the user is given a number of branching options. Additional variables, features, or areas can be selected, revisions to annotations, titles, or color ranges can be made, or various replay and output options selected. Branching is accomplished by using pushbutton fields within choice groups on AF program entry windows. SCL code within the program entry checks the users selection and calls the appropriate program entry. When the user branches back to another variable, feature or area selection, a CALL GOTO is used rather than a CALL DISPLAY so

that a large execution stack is not built up.

ANNOTATION

The annotation data set is used to add descriptive information to the maps, such as road numbers, county names, milepost limits, and project numbers. Default annotations are based largely on the type of area the user has selected. The annotation data set is created by processing the feature map data set and outputting annotation observations at appropriate map locations. Since roadway densities vary around the state, an overlap checking routine is used to eliminate annotations that may overlap another. The user is given a great deal of flexibility in overriding the default annotations as to size, types and allowable closeness. The capability is provided for the user to save his own sets of customized annotation settings for later use.

A separate module for adhoc maps has also been provided where the user can access his own feature data set that he has created. This data set must have been created using the generic feature response data set structure, but can contain data values of his own choosing. The user must initially specify all his titles, legends, data ranges and colors, but these can stored for later use. This module has proven useful for non-pavement applications by other offices, as well as for responding quickly to upper management requests for special maps.

OUTPUT AND REPLAY OPTIONS

During a PAVMARS session, all the graphics that a user generates are saved in a SAS work catalog. Options are given to the user to perform various functions on these graphics. A permanent SAS dataset can be allocated and selected graphics can be copied to this dataset using PROC GREPLAY. GREPLAY can also be used to redisplay graphics, either from the work catalog or from a permanent data set. Predefined template panels are also available for displaying multiple graphics on one screen, or to zoom to selected locations within a graphic. Template panels are selected by entering a panel number next to the

graphic description in the GREPLAY window. Zooming is accomplished by defining template panel dimensions greater than the screen display dimensions.

Various hardcopy output options are provided to the user. Plots can be routed to attached or remote plotters, or laser printers. An HPGL (Hewlett-Packard Graphic Language) format data set can also be allocated and created for a selected graphic. This data set can be downloaded to a PC or passed across networks for use on other system plotters or other graphic packages.

Setting SAS graphics hardware options can be confusing, particularly to novice users. To alleviate this, pop-up selection lists are provided for the user to select the type and location of the hardware device he wants to use. SCL code is then used to set the graphic options that are appropriate for that device.

An option to generate a report listing the data values for a generated map is also provided. Since a single response data set structure is used for all maps, only one program module is required for all feature options. Feature macro variables for titles and variable labels assigned by the extract modules are used to provide customized data descriptions on the printed report. Other specialized reports are also available from PAVMARS through other window selections. Branching to other SAS/AF applications without leaving SAS is also provided using the CALL GOTO routine. When branching between SAS/AF applications, proper allocation of files must be provided within the application and the naming of stored macros needs to be coordinated.

ZOOMING AND QUERYING

Though the PROC GREPLAY window provides the user with a great deal of capabilities for graphic manipulation, it can also be confusing to a new user. Another option for graphic redisplay has been provided in PAVMARS that utilizes the SAS graph window and PROC GREPLAY in line mode (non full-screen option).

With this option the user is presented a pop-up selection list to select the graphic entry for redisplay. An SCL

program is then used to open a graph window, display the selected graphic, resize the window, set autoterm to off, end the graph window without it closing, and return control to the user in an AF window. From this window, the user is given the options to zoom in or out, perform database queries, select another graphic or exit.

This window is particularly useful for examining detailed map information and reviewing database information about a project or section of roadway. The interactive zoom capabilities allow annotation sizes to be reduced during map generation and more information displayed on the map without causing annotation overlap problems.

The user can identify a section of roadway of interest by the color coding, then zoom to this area and determine the section or project identification number from the map annotations. With the map still displayed, a database query window can be opened and the identification number of the section of interest entered. The database information is retrieved and displayed in an extended table window next to the map window. The user can scroll through the extended table or enter additional identification numbers to query other sections of interest. Cross-linking between databases is also provided to allow the user to query different databases without having to reenter identification numbers.

The following code demonstrates how the zoom process is accomplished.

The selected graphic entry has already been copied to catalog templ and renamed as template. The name template is used since this is the default entry name when replaying a graphic to a template panel.

```
/* close graph window so that */
/* templ is freed */

call execcmdi('graph1;autoterm on;
end');

submit continue;
/* greplay in line mode(nofs) */

proc greplay nofs;
  igout=templ;
  gout=temp2;
```

```

tc=ztout;

/* define template panel 1 */
/* xlate and scale values set */
/* based on user selection */
/* and whether zoom in or out */

tdef ztentry
  1/def
    xlatex=&xlatex
    xlatey=&xlatey
    scalex=&scale
    scaley=&scale
  ;
  template ztentry;

/* replay templ entry named */
/* template using panel 1 */
/* to temp2 gout catalog */

treplay 1:template;
run;

/* delete templ entry */

delete template;
run;

/* swap catalogs and copy */
/* template back to templ */

igout=temp2;
gout=templ;
copy template;
delete template;

/* delete template entry */

tdelete ztentry;
run;
quit;
endsubmit;

/* display templ, resize, end */
/* autoterm off leaves open */

call execcmdi( 'graph1
work.templ.template;
wdef 1 1 18 50; autoterm off;
end');

```

The scalex and scaley variables control how much the template panel size is increased. The xlatex and xlatey variables control the translation of the panel in the x and y directions. When the user selects whether to zoom in or out and the location, the SCL program sets SCL variables &scale, &xlatex and &xlatey that are used by the submit

block code.

For example, to zoom in to the center at 2 power, the scale would be set to 2 and xlatex and xlatey to 0. The scale factor causes the template panel dimensions to increase from 0 to 100 to -50 to 150. The screen only displays from 0 to 100. To zoom to the upper left corner, xlatex would be set to 50 and xlatey to -50 to move the template panel 50% to the right and 50% down.

To zoom back out from the upper left corner, the scale factor is set to 0.5, xlatex to -25 and xlatey to 25. To keep track of how far a user has zoomed in, the new SAS 6.07 list structure is used as a stack to store zoom locations and powers. These entries are popped back off the stack when the user selects zoom out and used to define the zoom out template panel.

FUTURE DEVELOPMENTS

A number of additional modules are planned for PAVMARS to display new data sources and analysis results that will be made available for pavement management. This includes information such as pavement cores that are now being recorded in a database, and new technologies such as ground penetrating radar for pavement thickness that are being developed. As discussed previously, the modular structure of the system makes this a straight forward process.

The potential for distributing part of PAVMARS down to desktop workstations is also being evaluated. Successful processing of PAVMARS code on a PC using SAS for OS/2 has been accomplished with virtually no need for code changes. Access to mainframe data using SAS/CONNECT with various mainframe to workstation connection options is now being examined.

The introduction of SAS/GIS in version 6.09 is eagerly awaited with its potential for more interactive graphics and database querying. GIS products from other vendors have been looked at, but hardware limitations, costs, and software compatibility and training issues have not made them suitable for pavement management purposes at FDOT at this time. Having a GIS product that is part of the SAS Software System should alleviate many of these problems.