2002

NASA FACULTY FELLOWSHIP PROGRAM

MARSHALL SPACE FLIGHT CENTER
THE UNIVERSITY OF ALABAMA

DEVELOPING A MATLAB®-BASED TOOL FOR VISUALIZATION AND TRANSFORMATION

Prepared By:	Blake J. Anderton
Academic Rank:	Student (Undergraduate)
Institution and Department:	Lipscomb University Engineering Mechanics Department
NASA/MSFC Directorate:	Engineering
MSFC Colleague:	Ms. Kathy Kappus

Introduction

An important step in the structural design and development of spacecraft is the experimental identification of a structure's modal characteristics, such as its natural frequencies and modes of vibration. These characteristics are vital to developing a representative model of any given structure or analyzing the range of input frequencies that can be handled by a particular structure. When setting up such a representative model of a structure, careful measurements using precision equipment (such as accelerometers and instrumented hammers) must be made on many individual points of the structure in question. The coordinate location of each data point is used to construct a wireframe geometric model of the structure. Response measurements obtained from the accelerometers is used to generate the modal shapes of the particular structure. Graphically, this is displayed as a combination of the ways a structure will ideally respond to a specified force input.

Two types of models of the tested structure are often used in modal analysis: an analytic model showing expected behavior of the structure, and an experimental model showing measured results due to observed phenomena. To evaluate the results from the experimental model, a comparison of analytic and experimental results must be made between the two models. However, comparisons between these two models become difficult when the two coordinate orientations differ in a manner such that results are displayed in an unclear fashion. Such a problem proposes the need for a tool that not only communicates a graphical image of a structure's wireframe geometry based on various measurement locations (called *nodes*), but also allows for a type of transformation of the image's coordinate geometry so that a model's coordinate orientation is made to match the orientation of another model. Such a tool should also be designed so that it is able to construct coordinate geometry based on many different listings of node locations and is able to transform the wireframe coordinate orientation to match almost any possible orientation (i.e. it should not be a "problem specific" application) if it is to be of much value in modal analysis. Also, since universal files are used to store modal parameters and wireframe geometry, the tool must be able to read and extract information from universal files and use these files to exchange model data.

The purpose of this project is to develop such a tool as a computer graphical user interface (GUI) capable of performing the following tasks:

- Browsing for a particular universal file within the computer directory and displaying the name of this file to the screen.
- Plotting each of the nodes within the universal file in a useful, descriptive, and easily understood figure.
- Reading the node numbers from the selected file and listing these node numbers to the user for selection in an easily accessible format.
- Allowing for user selection of a new model orientation defined by three selected nodes.
- Allowing the user to specify a directory to which the transformed model's node locations will be saved, and saving the transformed node locations to the specified file.

Program Development

For this program, MATLAB 6.1 was chosen as the software most capable of generating the desired transformation results within a graphical user interface. A graphical user interface (GUI) is defined as a user interface built with graphical objects, such as buttons, text fields, sliders, and menus.

The programming process first began by sketching the most important attributes of the interface on paper to get an idea for what the finished program would resemble. Steps were taken to make sure every step necessary to achieving the project purpose were reflected in these design sketches. Using MATLAB 6.1's Graphical User Interface Development Environment (GUIDE) software, the following GUI figure was then designed as the program's graphical user interface.

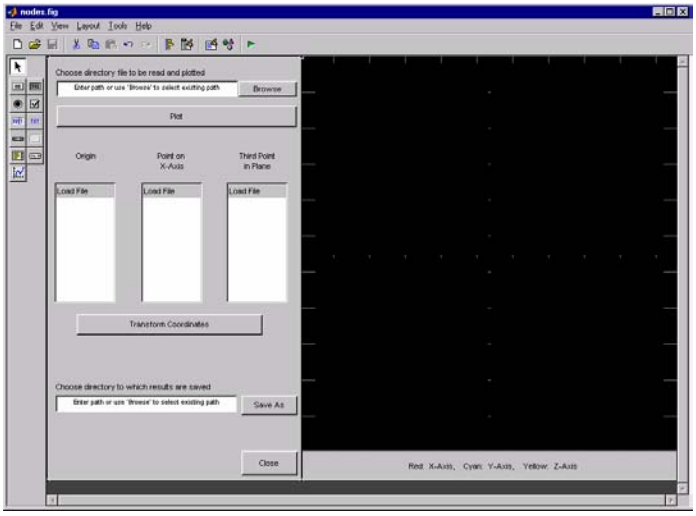

Figure 1: Layout of Program User Interface (using MATLAB 6.1 GUIDE)

As seen in Figure 1, the program was designed to be able to read a universal file, plot the unaltered geometry on the set of axes (in the darkened right half of the screen), extract and display a list of node numbers (in the three parallel, vertical listboxes), allow the user to select which three nodes constituted the transformation x-y plane by clicking nodes in the appropriate listboxes, transform the geometry to match the new x-y, and store the transformed geometry in a specified universal file.

One important note concerning the program coding between versions of MATLAB: while the code in the program was designed for use in MATLAB 6.1, it cannot be used in earlier versions of MATLAB (such as version 5.3). MATLAB's application m-files are designed to support a previous version's applications but will not support a later version's applications (i.e. code from version 6.1 will not run on version 5.3). The problems encountered when switching between different versions constituted much of the time used in program development (since the transformation code was developed using version 5.3).

Mathematical Theory

To show the mathematical basis for the nodal coordinate transformation, consider the figure below:

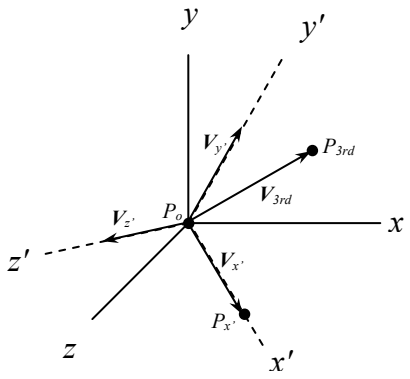

Figure 2: Example Coordinate transformation (from *xyz* to *x'y'z'* system)

Note that, unlike the above figure, the new origin does not have to have the same location as the original. The goal in transformation is to find matrix [*T*] that transforms coordinate locations defined by unit vectors in the *xyz* plane (denoted as [xyz]) to locations defined by unit vectors in the *x'y'z'* plane (denoted as [*x'y'z'*]). This is expressed as:

$$[T][xyz] = [x'\,y'\,z'] \qquad \textbf{(1)}$$

Note that unit vectors for the transformed plane are shown as $\boldsymbol{V}_{x'}$, $\boldsymbol{V}_{y'}$, and $\boldsymbol{V}_{z'}$. Finding [*T*] becomes much easier when unit vectors define point locations because the right hand side of the previous equation is simply the identity matrix (since each component has a magnitude of 1 in one of the three coordinate directions). The transformation matrix is then found as follows:

$$[T] = [xyz]^{-1} \qquad \textbf{(2)}$$

Also note that to find the unit vectors, the only points needed are the new origin P_o, a point $P_{x'}$ to define the *x'* axis, and a point P_{3rd} in the *x'-y'* plane. From these points, the vectors mentioned above can be found using cross products. The program is designed to give the user the option of choosing these three specific points from node locations.

Results

With the program appropriately designed coded to meet the previously stated objectives, testing was done using a number of sample models. This produced new ideas on program improvement and user accessibility. Features such as a black axes background and other details were added and error messages were designed to keep the user from making mistakes. Universal files with coordinates from the Italian heat exchanger tested by engineers from the Modal and Control Dynamics team at Marshall Space Flight Center were used in testing. The following figure

shows pictures of the program when first generating the plot (left) and after activating the transformation (right). Notice that the filename and filepath of the figure is displayed in both of the edit boxes (the features with white backgrounds on the left of the screen). This figure also shows that a top-front-side view is displayed along with an isometric view of the part's geometry.

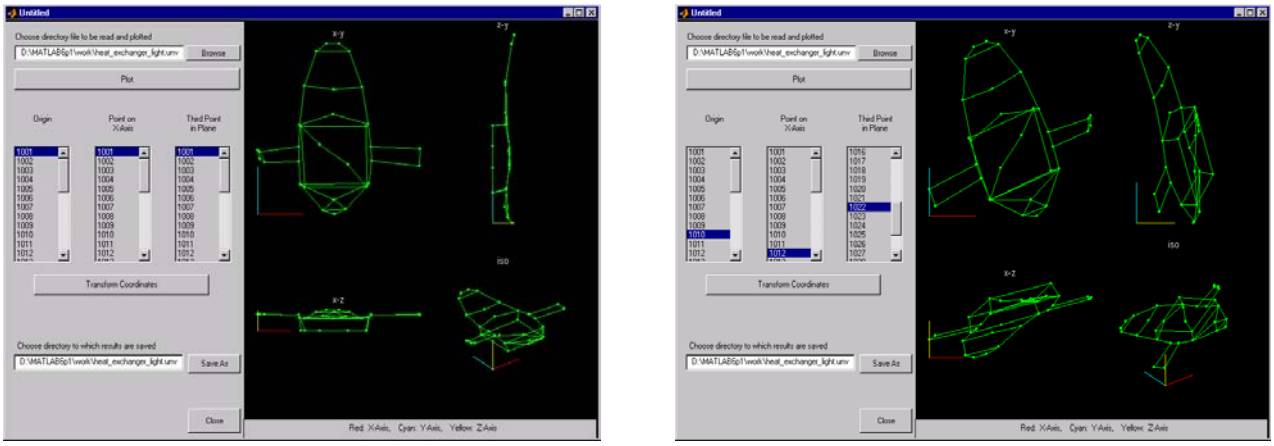

Figure 3: Display of unaltered geometry (left) and transformed geometry.

Future Work and Applications for the Coordinate Transformer

The program successfully displays structure geometry and allows coordinate transformation along with an updated display of the transformed figure. The program also allows for rewriting data to stored universal files as well creating new universal files in which to store transformed (or untransformed) node coordinates. However, additional work can be done to make this program more applicable to modal analysis. A list of possible additional work to be done on the project is as follows:

- A feature showing the mode shapes can be added so that the analytical response of the structure in question can be observed directly from the universal file without having to transfer the file to another application in order to view mode shapes.
- A feature can be added to bring up a figure showing node numbers directly beside the nodes they represent.
- Other features such as right-click menus can allow for user display preferences.

References

[1] Ewins, D. J. (2000), *Modal Testing: Theory, Practice and Application, 2nd ed.*, Research Studies Press Ltd., pp. 1-23.

[2] *MATLAB® Online Help* (2002), "Using MATLAB: Creating Graphical User Interfaces", http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml