COMPUTER-AIDED DESIGN & APPLICATIONS, 2016
VOL. 13, NO. 2, 184–198
http://dx.doi.org/10.1080/16864360.2015.1084185

Application of machine learning and parametric NURBS geometry to mode shape identification

Robert Porter, Ammon Hepworth and C. Greg Jensen

Brigham Young University, USA

ABSTRACT

In any design, the dynamic characteristics of a part are dependent on its geometric and material properties. Identifying vibrational mode shapes within an iterative design process becomes difficult and time consuming due to the frequently changing part definition. Although research has been done to improve the process, visual inspection of analysis results is still the current means of identifying each vibrational mode determined by a modal analysis. This paper investigates the automation of the mode shape identification process through the use of parametric geometry and machine learning. This allows the designer to gain a more complete view of the parts' dynamic properties. It also allows for increased time savings over the current standard of visual inspection.

KEYWORDS

Mode shape identification; machine learning; parametric geometry

1. Introduction

Identifying vibrational mode shapes with their corresponding frequencies becomes important when designing objects or structures that are subjected to dynamic forces. That is, to alleviate structural weakness due to resonant behavior or natural frequencies being excited by operational forces [8]. As a check, the designer can perform a modal analysis using the Finite Element Method to easily identify the mode shape of his/her given object. However, in optimization, where the design changes on an iterative basis, identifying and comparing these mode shapes becomes a complex problem. In 2011 Selin et al. explored applying parametric NURBS geometry and the Modal Assurance Criterion (MAC) to mode shape identification. The result of this research showed that automatically identifying mode shapes of parts with differing geometries and mesh densities is possible [16].

This paper applies machine learning to the mode shape identification problem in efforts to improve the task of identification. Machine learning is broadly defined to include any computer program that improves its performance at some task through experience [14]. Machine learning has been used in various classification and regression problems where one may wish to know a type or category that given inputs fall under (classification) or a numerical prediction given inputs with some training data (regression) [1]. This paper seeks to leverage machine learning along with parametric NURBS geometries to classify vibrational mode shapes from a finite element analysis. In doing so, a designer can run an iterative optimization with information about the dynamic behavior of the object. While Selin's research was successful within its scope, automating the identification process utilizing machine learning will show added benefits and decreased disadvantages over utilizing the MAC. One of these added benefits is increased accuracy. The MAC based identification program developed by Selin had limited accuracy, and was not 100% accurate over the tested geometries [16]. This paper presents a method by which machine learning, along with parametric geometries will be used to automatically identify vibrational mode shapes and frequencies from displacement data. The described method will be an important step towards the development of a complete iterative vibrational mode shape identification tool that could be used for a wide variety of parts or models.

Within a design process modeled parts often go through many changes, especially within an iterative design process such as an optimization or design of experiments. During an iterative design process parametric models can be updated and thus change in either small or large ways. The purpose of changing model parameters and geometries, especially in an iterative design, is to obtain a design that is superior to an initial or starting design. Changing the properties of a part via geometry,

CONTACT Robert Porter rmcporter@gmail.com

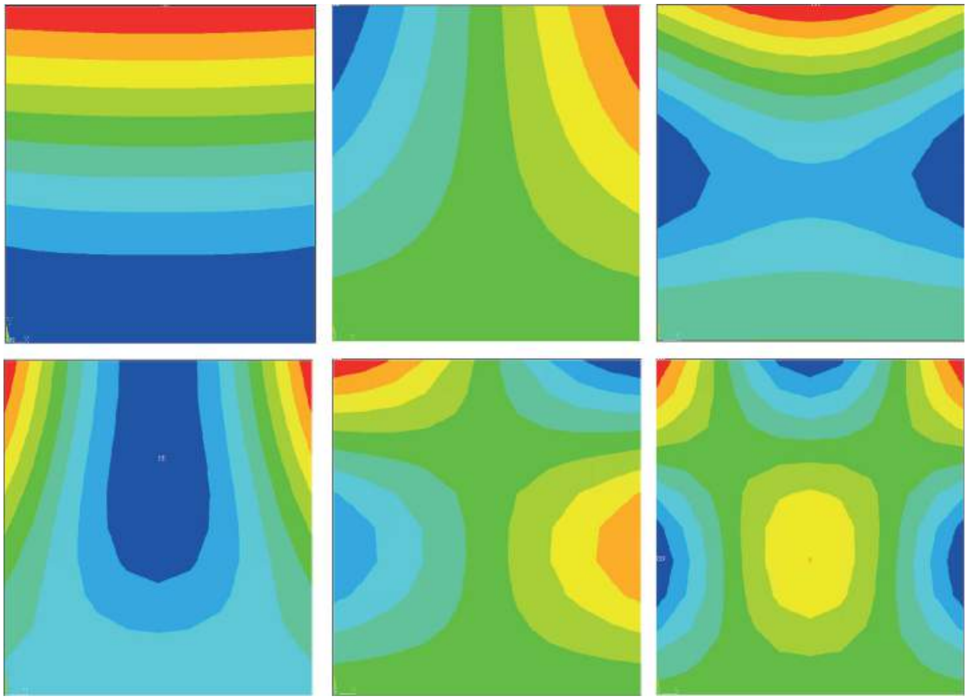

Figure 1. Mode shapes produced by a modal analysis.

material properties or other, can have a significant effect on its static and dynamic behavior.

In a modal analysis the natural frequencies are affected by design changes and the vibrational mode shapes excited by these natural frequencies can exhibit themselves in a different manner than in consequent design iterations. These results can be reported in order of increasing natural frequency. An example of the resulting contour plots from a modal analysis can be seen in Fig. 1. These contour plots are simple examples of what a designer would see if visually inspecting the results of a modal analysis. While what these results mean is only important to the designer's application, it is important to note that the mode shapes are affected by design changes and the vibrational mode shape excited by the parts natural frequency can exhibit themselves in a different sequence on different design iterations. This makes the task of identifying the specific mode shape associated with each natural frequency more complicated when executing an iterative design. If a designer was to view these changes after each iteration step, they may take measurements of the analysis results or simply view the surface to see how the mode shape has changed. In this way the designer identifies the mode shape like they would any other shape, by looking at the objects features (number of sides, scale, how many peaks and valleys and their locations on the part, the angles contained within the part, etc.). The designer can identify the mode shape without regard to its mesh density or overall size. Machine learning has the ability to take in attributes, like those described, and given a label can create its own hypothesis as to the correlation between the two [12]. Machine learning has been used in a wide variety of classification problems where the inputs and outputs of a system for some instance are known but how to arrive at the output is unknown or unclear. Applying machine learning to mode shape identification is not a straight forward process. Some conditioning must take place in order to benefit from machine learning capabilities. This conditioning will include identifying points on the given surface and generating a NURBS surface to use in the training of the machine learning algorithm. This is discussed in detail in section 3.2. Once pre-processing is done, an algorithm must be chosen and that algorithm must be trained given training examples thus allowing it to learn. It will be shown that a properly trained machine learning algorithm is able to classify mode shapes with good accuracy.

2. Background

2.1. Modal analysis

In a design process, a modal analysis becomes useful when the designer wishes to characterize the dynamic behavior of a part or structure. The dynamic characteristics that the modal analysis determines are the natural frequencies and the vibrational mode shape of the given part or structure [8]. Finite Element solvers perform modal analysis on meshed models with results including a natural frequency and a vector of nodal displacements (mode shape). Historically the MAC has been the general method for measuring the consistency of modal vector estimates, calculations, or experimental data [4]. The MAC has the ability to calculate the linearity between two modal vectors. After running the MAC calculation, it results in a value between zero and one, which indicates the correlation between the two mode shapes, one being a perfect match and zero being no correlation. One of the downsides to using the MAC is the inability to compare modal solutions from a Finite Element model that has differing meshes or number of elements. The usefulness of the MAC has further been described in Selin's work [16]. In Selin's research he was able to leverage parametric NURBS geometries in conjunction with the modal assurance criterion to automatically identify vibrational mode shapes and frequencies from modal analysis displacement data. This work will be benchmarked against this work with the hopes of creating a more robust method. For more information on the MAC, see [8, 6, 12].

2.2. Machine learning

Machine learning is actively being used today in many instances where it is necessary to turn data into information. Machine learning is a mix of computer science, engineering, and statistics, and often appears in many other disciplines from politics to geosciences [10]. Any field that needs to interpret and act on data can benefit from machine learning techniques. While machine learning has been successfully applied to many classification problems in many fields, there is no known research that has leveraged machine learning to identify vibrational mode shapes. To properly implement machine learning for this paper and to any problem that can benefit from machine learning some design decisions must be made. These decisions include identifying the type of knowledge to be learned, how to represent the target knowledge, and a learning mechanism [14]. For more information, see [1, 3, 14, 18].

2.3. Parametric geometry

Parametric geometries are beneficial in that they are quick and easy to compute (x,y), or (x,y,z) coordinates of points existing on a curve or surface [15]. By generating a parametric surface with a uniform parameterization, surfaces of differing size and complexity can be segmented into distinct areas through using a u and v coordinate system separate from the actual x,y,z coordinate system. Non-uniform rational B-spline (NURBS) surfaces are widely used parametric geometry representations. Each surface can be parameterized such that u-values of zero and one correspond to two edges of the surface and v-values of zero and one correspond to the other edges of the surface as shown in Fig. 2. Since NURBS surfaces are defined this way, they can only be used to represent surfaces that are four-sided, which can be a problem. The benefit of using parametric geometry is that surfaces of varying size and shape can relate to a single set of attributes by means of similar parameterizations that can directly relate the points on each surface. This becomes important when assigning feature attributes that must remain consistent through a variety of surfaces that can vary in size, shape, and mesh densities.

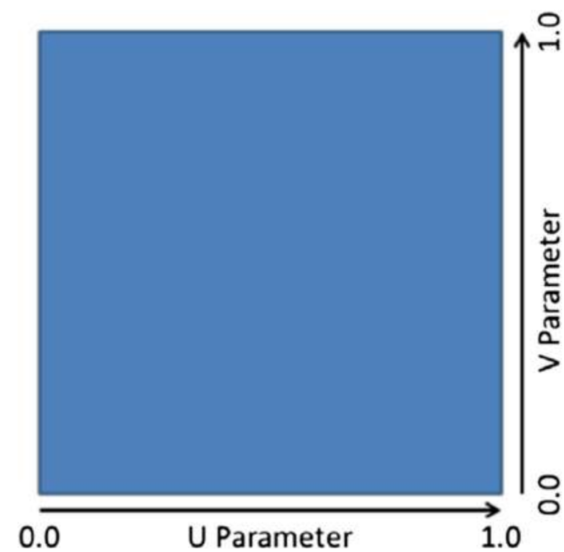

Figure 2. A parameterized surface.

Interpolation through existing points and extrapolating points from control points are common methods for creating NURBS surfaces [20]. This paper will utilize global surface interpolation to create NURBS surfaces from point clouds which approximates a surface through a set of existing points. In creating a NURBS surface, each control point defining a surface has a weight and a B-spline basis function which together determine the extent to which the point influences the surface. A knot vector in each parametric direction, u and v, influences the surface topology as well. The following equation 2.1, is the mathematical definition of a NURBS surface of degree p in the u direction and degree q in the v direction. More information on NURBS see [15, 20]. NURBS surfaces have also been used as basis functions for the finite element method as seen in the isogeometric analysis [7, 17].

$$S(u,v)=\frac{\sum_{i=0}^{n}\sum_{j=0}^{m}N_{i,p}(u)N_{j,q}(v)W_{i,j}P_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m}N_{i,p}(u)N_{j,q}(v)W_{i,j}},\quad 0\le u,v\le 1 \tag{2.1}$$

3. Method

We present a method of automatically identifying the mode shape of an object represented by a NURBS surface resulting from a modal analysis by using machine learning. To achieve our goal a classifier must be obtained. A classifier is a function that can identify to which category a new observation belongs. For example, a classifier could be trained to identify between spam and non-spam email messages. After learning, it can then identify if an incoming email is spam or not and take an appropriate action. Once trained, our mode shape classifier can be utilized in a stand-alone or in an automated approach to mode shape identification. The following steps outline this method:

1. Gather information from the designer in regards to a base line design and the design space.
2. Normalize, transform, and label nodal displacements resulting from a modal analysis into a NURBS surface representation in preparation for machine learning.
3. Obtain and evaluate a classifier that will be used to identify the mode shape for a given part.
4. Use the classifier to automatically match and report vibrational mode shapes.

3.1. Gather user information

Some information from the designer is required before training or testing the method. A designer must specify the model parameters that will act as a baseline design. These parameters represent the dimensions or properties of the part such as length, width, thickness, mesh coarseness, etc. The parameters of the baseline design that will be changed or iterated upon would be those that would either optimize some aspect of the model or fulfill some requirement of an overall design. As model parameters change new designs will arise that may not have been observed previously and may result in unexpected mode shapes.

3.2. Data pre-processing

Below describes the method used to prepare data resulting from a modal analysis by normalizing and transforming the resulting nodal displacements into a mathematical NURBS surface representation. A modal analysis is done on a Finite Element model of a part or component and the results are written to a file. This file contains the position and displacements of every node in the model corresponding to its natural frequency solution. This method parses through this file to obtain desired information.

In order for the method to obtain consistent comparable attributes on which to consistently identify a given mode shape, all surfaces are normalized by the maximum nodal displacement found in the solution set. For the normalizing equation below, U_{all} denotes the set of all nodal displacements in the model solution, therefore the normalized displacements, U_{norm}, are found by:

$$U_{norm}=\frac{U_{all}}{\max(|U_{all}|)} \tag{3.1}$$

This results in normalizing all nodal displacements falling between numerical values of negative one and positive one. In Eqn. (3.1) the max function finds the largest absolute value contained in U_{all}.

When creating a NURBS surface, the normalized positions and displacements for each node are used. A global surface interpolation fits the data to a preliminary surface through a set of points containing the normalized node locations from the original model. This surface is projected onto a working plane (see Fig. 3). The red, green, and blue dots in Fig. 3 represent points on the surface as they are projected onto the working plane.

The parameter values associated with every node in the model are determined by querying the surface and are stored for future use. A new point set is then created with data from each node. Points in this set are defined in three dimensions by the u and v parameters of the node on the preliminary surface and the normalized displacement of the node that is perpendicular to the working plane. The surface is fit again with this new point set, creating a new surface representing the mode shape (see Fig. 4).

After a mode shape surface is created we then can make comparisons with other surfaces in the presence of differing part definitions such as length, width, thickness, mesh coarseness, etc. An example of this can be seen visually in Fig. 5 where the top level of three surfaces have the same mode shape but result from differing geometries. In this example, the original geometries consisted of varied edge lengths, and widths as well as thicknesses. Each surface was then normalized and projected to a working plane as described above. After these surfaces were fit with the displacement data that resulted from the modal analysis, we can compare them. For illustration purposes, three similar modes shapes were chosen. One can visually determine how close these surfaces relate despite differences in their original part definitions.

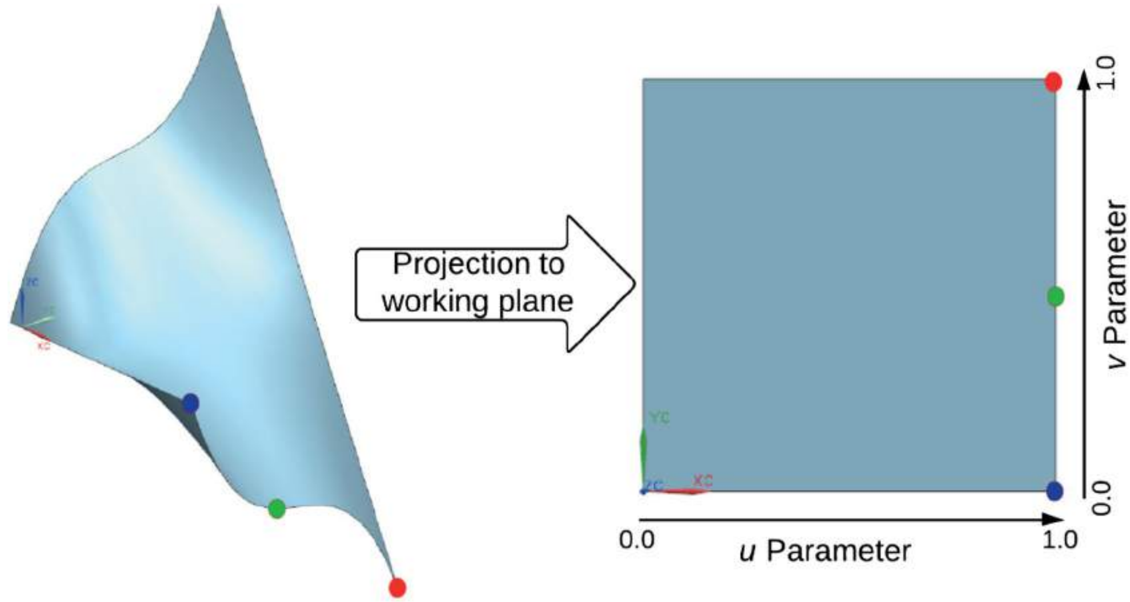

Figure 3. Normalized surface parameterized and projected onto working plane.

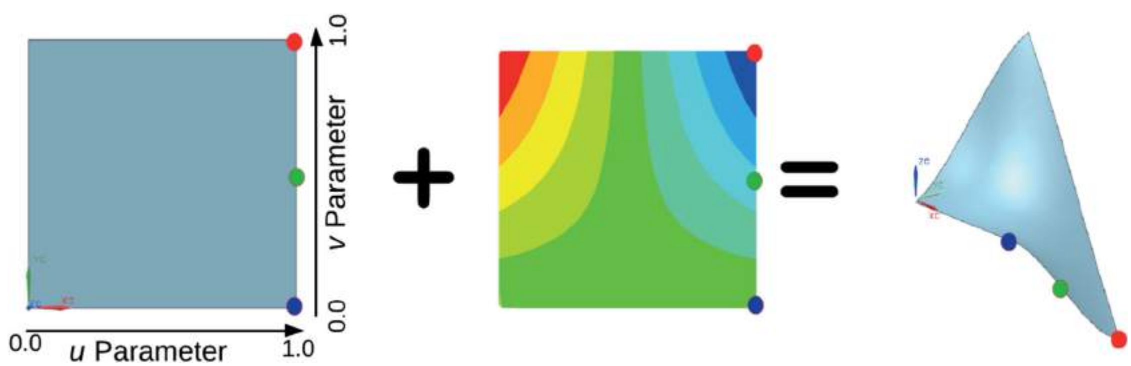

Figure 4. The preliminary surface and displacement data create the mode shape surface.

Once a surface has been created to represent the model's mode shape, displacements are queried from that surface. The parametric surface displacements in the z direction are grouped together in a grid like fashion. Each section of the grid represents the average displacement found over a given number of nodes. How these displacements are grouped is shown in Eqn. 3.2.

$$\bar{x}_z = \frac{\sum_{u=0}^{M} \sum_{v=0}^{N} Q_{u,v}(z)}{n} \tag{3.2}$$

In this equation $Q_{u,v}$ is the vector containing all nodal displacements. x_z is the averaged displacement found over the number of nodes per grouping n. Each average displacement is considered a feature attribute of the surface. All average displacements are then written as a geometric definition of the surface to a file in preparation for machine learning. For machine learning a set of geometric definitions that describe a surface are called instances, and each instance contain feature attributes. Again, it is important to note that the number of feature attributes must remain consistent throughout this process for a basis of comparison as mentioned previously.

An example of how these geometric representations of mode shape surfaces are transformed into instances for machine learning can be seen in Fig. 6. This illustrates points that have been identified, grouped, and averaged into sections to act as features of the surface. The number of points and thus the number of sections required for this method is dependent on complexity. More complex geometry and/or higher order mode shapes would require more features to adequately describe the mode shape.

3.3. Machine learning

This section reviews the method used for obtaining a classifier that will be used in this research to identify the mode shape for a given part. The first step toward

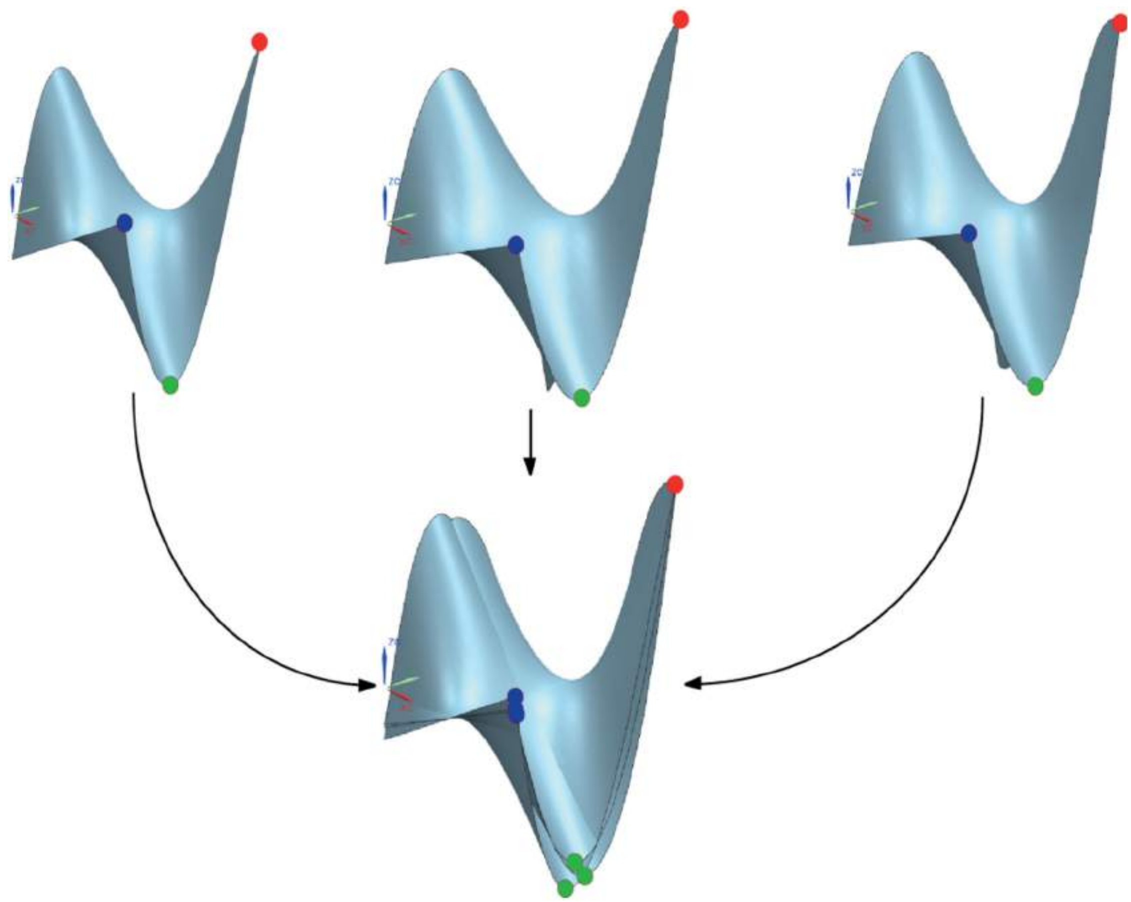

Figure 5. Normalization of geometry.

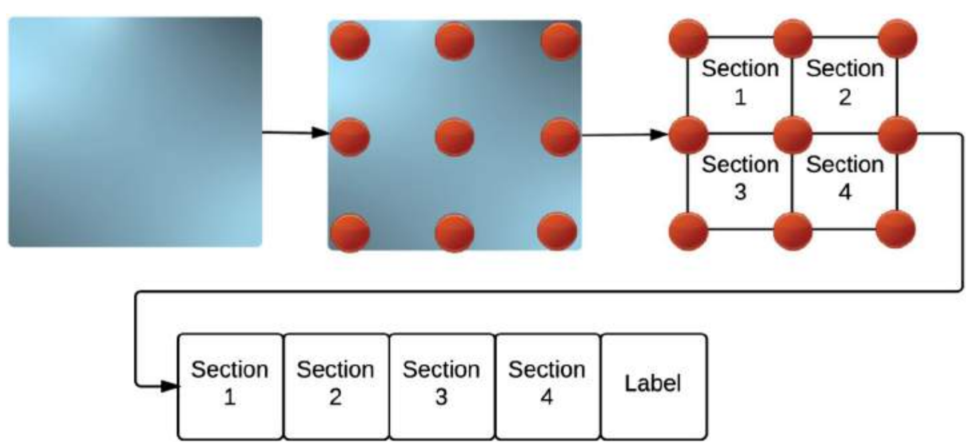

Figure 6. Grouping average displacements into instances.

obtaining a classifier is collecting and defining a training dataset. A training set contains examples that are characteristic of the problem to be solved. Every geometric definition of a surface, or in other words, every instance from the previous data pre-processing step must be stored with a unique user-defined label for its mode shape. Upon creating these training instances, they are stored for further training and testing where they may be adjusted. This training set may be adjusted to contain more examples or different feature attributes if it is found that the attributes chosen are insufficient.

For example, in order to collect a training dataset to use in classifying an animal as mammal or reptile, a list of unique traits would be generated to describe the characteristics of the animal to be identified. These characteristics could include warm or cold blooded, has fur, lays eggs, etc. A string of characteristics describing a mammal would be labeled 'mammal' at the end of the string. After a training dataset contains a number of examples of each animal group, an algorithm can then be used to learn to classify animals based on the examples given.

Choosing a specific learning algorithm to use in this classification problem is a vital step. It is known in machine learning that there is no algorithm that is uniformly superior over all possible problems [19]. There are several supervised machine learning algorithms to choose from. The optimal algorithm is determined by cross validation to ascertain how well the algorithm can learn the training data and by a paired t-test. Cross validation is a statistical method of evaluating and comparing learning algorithms by dividing data into segments where one is used to learn or train a model and the other used to validate the model. The basic form of cross-validation is k-fold cross validation where the data may be divided in a specific number of segments [1, 14]. A t-test is a common method that assesses whether the algorithms chosen are significantly different from each other in the presence of a supplied training set. Given two paired sets (classification results) of n measured values, the paired t-test determines whether they differ from each other in a significant way under the assumptions that the paired differences are independent and identically normally distributed [2]. After an appropriate classification algorithm is selected based on the results from cross validation and a paired t-test, we can then use it to run the method. While there are a number of algorithms to choose from, this research implements k-Nearest Neighbors (k-NN), Decision Trees, and Support Vector Machines (SVM). Please refer to the following references for more information on these algorithms [1, 5, 9]. While there are various algorithms to choose from, these three will give the reader a good idea of how this method performs and how well we chose our feature attributes.

Exposing classification algorithms to a training set allows it to learn to do its task. It learns by looking at the provided examples of attributes and given a label or class can create its own hypothesis as to the correlation between them [12]. By training our classifier we will be evaluating how well the learning algorithm is able to learn the sample training data. Many classification algorithms allow for some tuning of parameters which can enhance the classifiers ability to interpret given data. If it is found that an algorithm is not performing well, some parameter tuning of that algorithm may increase its ability to learn.

When applying this method to a new design space, a designer may evaluate the algorithm chosen to ascertain how accurately that algorithm is at predicting mode shapes. To evaluate our chosen classifier, we will use test sets where every instance has been properly labeled with its corresponding mode shape. These test sets contain instances of mode shapes that result from parts or models that may or may not have been observed when creating the training data. For example, training data may be compiled from a model that has varied in its parameters by $\pm$ 10% and testing data may be compiled from the same model that has varied in its parameters from $\pm$ 10% to $\pm$100%. While the change in parameters are randomized every iteration, there is a chance that within $\pm$ 10% some test instance may be identical to some training instance. By testing on data that has not been previously observed we can evaluate how well the classifier and this method can extrapolate a mode shape from geometry with which it has no previous experience. If the training or evaluation of a classifier is unsatisfactory the designer can return to a previous stage of the supervised machine learning process. For the problem of mode shape identification a number of factors can be investigated: the most relevant features may not be taken into account, a larger training set may be needed, and utilizing an inappropriate algorithm or some parameters need tuning.

3.4. Mode shape identification

Once a classifier has been obtained we can now use it to assign class labels (mode shape names) to instances where feature values are known but the class is unknown. This can be done both within and without an optimization workflow. Upon obtaining a classifier a new sequence is needed for implementation of this method. While the first 2 steps remain the same, gathering information from the user and data pre-processing, we will now use the classifier directly to label mode shapes with the accuracy of the selected classifier.

4. Implementation

The methods were implemented in computer programs in order to be utilized in an automated approach where a design can be iterated upon, or as a standalone process where the designer can examine a specific design. A program was developed to create NURBS surfaces to represent mode shapes for a set of modal analysis results. This program outputs discretized displacements of surfaces to be used as feature attributes in the mode shape identification process. This uses Siemens NX Open API for the fitting algorithm and ANSYS Parametric Design Language for the automation of the modeling, meshing, analysis, and results reporting for the training and testing. All of the modal analysis completed for this research was done using ANSYS 13.0. Weka 3.6.9 was used for its collection of machine learning algorithms and

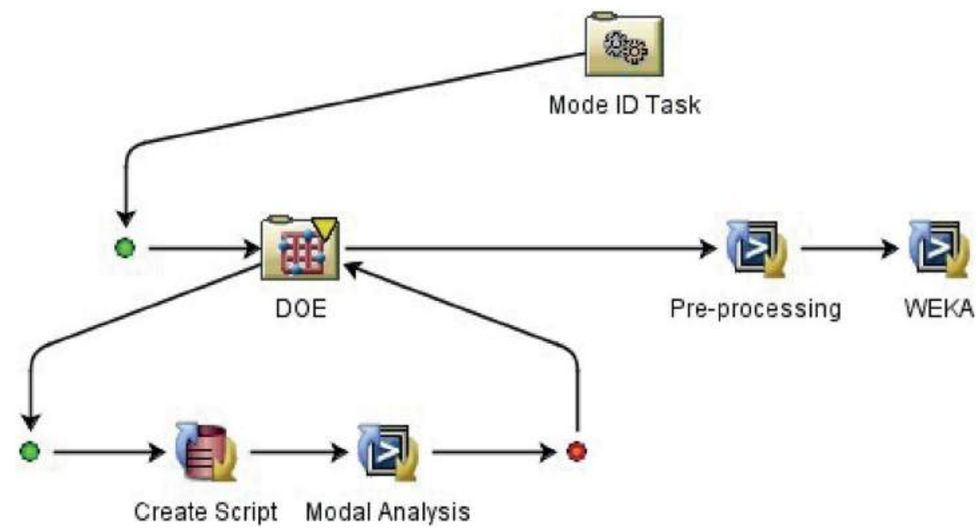

Figure 7. Isight task example.

ability to perform statistical analysis. The applications are integrated into SIMULIA's Isight optimization software to enable a designer to identify mode shapes iteratively or on a specific design see Fig. 7.

4.1. Gather user information

The task allows users to identify which parameters of a base design they wish to change, by how much, and how many times within the training DOE component. Once the user has specified their desired design space the task can then be ran for training in order to obtain a classifier or testing if a classifier has previously been obtained.

4.2. Data pre-processing

Preparing the data contained in the modal analysis file to become feature attributes has several steps. The modal analysis results which contain node locations and displacements are read and stored by the program. These results are normalized to properly compare and contrast parts with differing geometries the magnitudes of the nodal displacements. A parametric surface is then created using a NX Open API function to fit the data (see Fig. 8).

The resulting parametric surface can then be used to obtain information about the mode shape without regards to changing parameters in the design space by conforming all possible geometries to a uniform basis for comparison. That is, all possible geometries within the design space will be re-parameterized in terms of u-values and v-values. After creating the parametric surface that represents a mode shape, points in real space are retrieved from that surface. The displacements in the z direction are then placed in discrete groups as shown in Fig. 9. Each square in Fig. 9 represent the average displacement of the surface for that region.

4.3. Machine learning

The training and testing of algorithms to be used as classifiers is performed using Weka. In order to obtain a classifier that will properly identify the mode shape for a given part we first must train and test an algorithm for classification.

We construct a training set by changing the baseline design of a model by $\pm$ 10%. For example, if a baseline design has a length of 10 inches the training set would include designs with lengths from 9 to 11 inches. After constructing the training data the designer must then

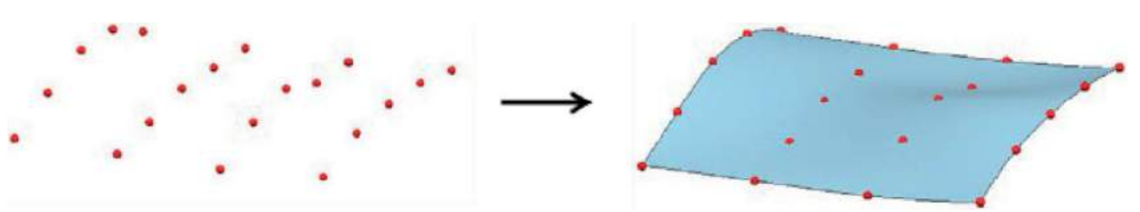

Figure 8. Interpolation of node data into a NURBS surface.

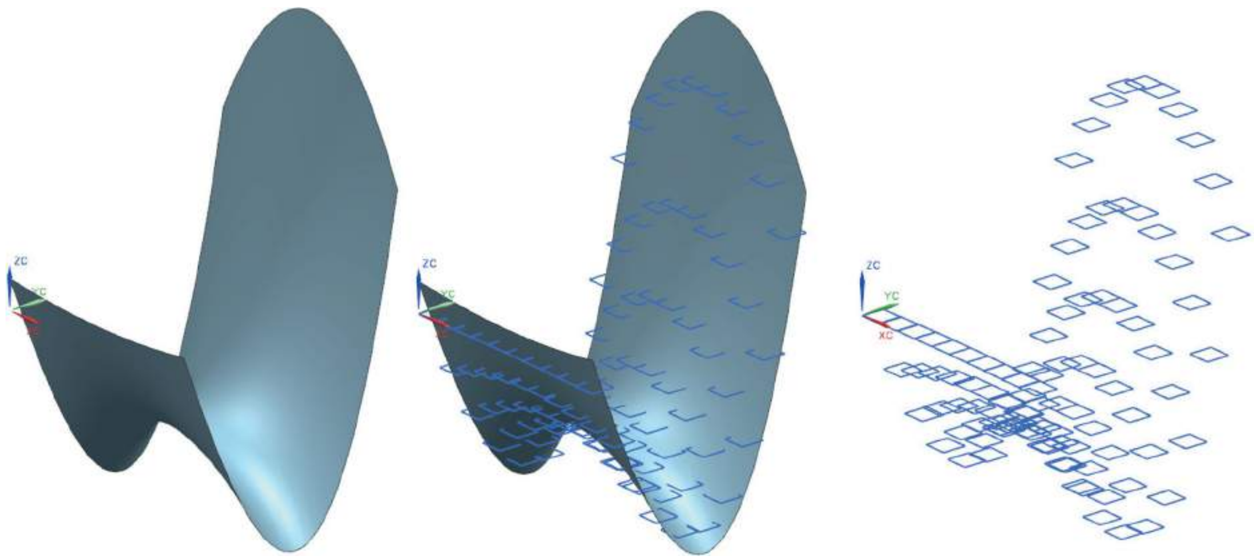

Figure 9. Discretization of a mode shape surface into sections.

label the instances in the ARFF file manually. To correctly label these mode shapes the designer may view each parts corresponding contour plot that results from the modal analysis. The number of instances in the training set is another consideration when looking at how well an algorithm can correlate the data with the given labels. The training set can be appended to through consequent runs in order to create a larger training set if needed. A larger training set may be desired if more training examples are needed to more accurately and consistently identify mode shapes.

Three algorithms to apply to this classification problem are chosen for machine learning algorithms: k-Nearest Neighbors (k-NN), Decision Trees, and Support Vector Machines (SVM). These algorithms are available in Weka's Explorer application [13]. In Weka the IBk algorithm was chosen which implements the k-NN approach as discussed. To implement a decision tree learner, Weka's J48 and Random Forest were chosen. J48 is a simple decision tree, whereas Random Forests operate by constructing a multitude of decision trees and outputting the class that is the mode of the classes output by individual trees. Weka's SMO algorithm was chosen to implement the SVM approach.

The training of each algorithm was also done using Weka's Explorer application with the training set obtained from the previous pre-processing step. Leave-One-Out Cross Validation (LOOCV) was used to evaluate how well each algorithm was able to learn the training data. LOOCV is a special case of the general k-fold cross validation method. LOOCV works by taking a data set with n examples and performs n experiments. LOOCV uses n-1 examples for training and the remaining example for testing. The overall accuracy can be obtained by averaging the accuracies computed on each experiment. LOOCV is used to allow for sparse training data so as to train on as many examples as possible.

Algorithms are then analyzed using Weka's Experimenter which enables users to create, run, modify, and analyze experiments [13]. To ascertain if there were statistical differences between the selected algorithms the Experimenter's paired t-test was used and the results of which can be found in Section 5.

In order to ascertain the robustness of the method, a learning algorithm is tested by subjecting the classifier to models that contained changes in their baseline design ranging from $\pm$ 10% to $\pm$ 100%. In this way we see how far this method can extrapolate outside of the initial training examples given. These tests were evaluated in Weka and the results of the training and testing of these algorithms will be given in Section 5.

5. Results

Various tests are performed to verify that the implementation of the method improves on previous automated attempts. While there are a number of modes that may be of interest to a designer, the tests looked at eight modes that result from a modal analysis for identification. These modes can be seen in Fig. 10. Each of these modes signifies their own class by being labeled as "Mode1", "Mode2" ... "Mode8". Any mode that cannot be classified as one of these modes is labeled as "Junk". It is the task of the classifier to distinguish between the different classes given the displacement attributes provided.

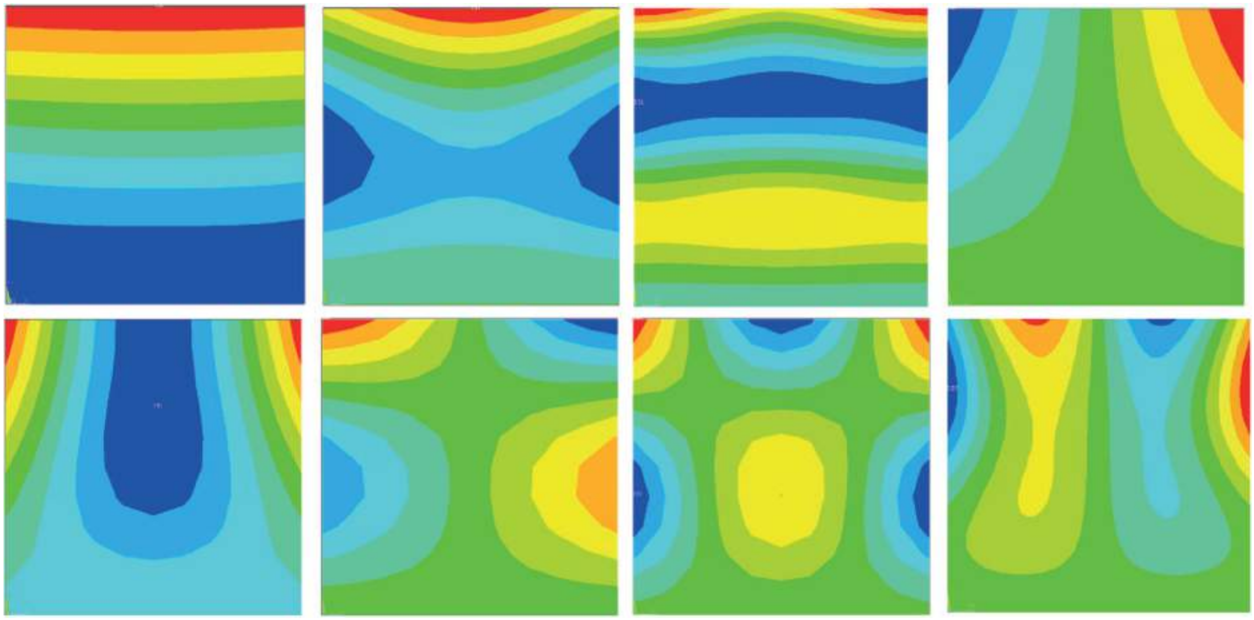

Figure 10. Eight trained mode shapes for identification.

5.1. Machine learning - algorithm selection

The method was tested on three different geometries. First a simple rectangular plate defined by four parameters: length, width, thickness, and mesh coarseness. The second was tapered and twisted along with changes in length, top width, bottom width, thickness, twist, and mesh coarseness. Lastly a plate was designed similar to the second plate with the addition of two nonlinear edges. A sample of these geometries can be seen in Fig. 11.

A training dataset of 200 instances was compiled containing variations in a model's baseline design of $\pm$ 10% as described in Section 4.3. The simple rectangular plate described above was the only model used to compile this training set. The training set containing 200 instances consists of 20 instances for each class "Mode1" through "Mode8" and 40 instances for the "Junk" class. 100 features were pulled from each surface to create each. This training set was used in the training and evaluation of each algorithm selected. How an algorithm performs can be seen visually in a layout known as a confusion matrix. For an example, how the k-NN algorithm performs with the training set can be seen in the confusion matrix in Table 1 where LOOCV was used.

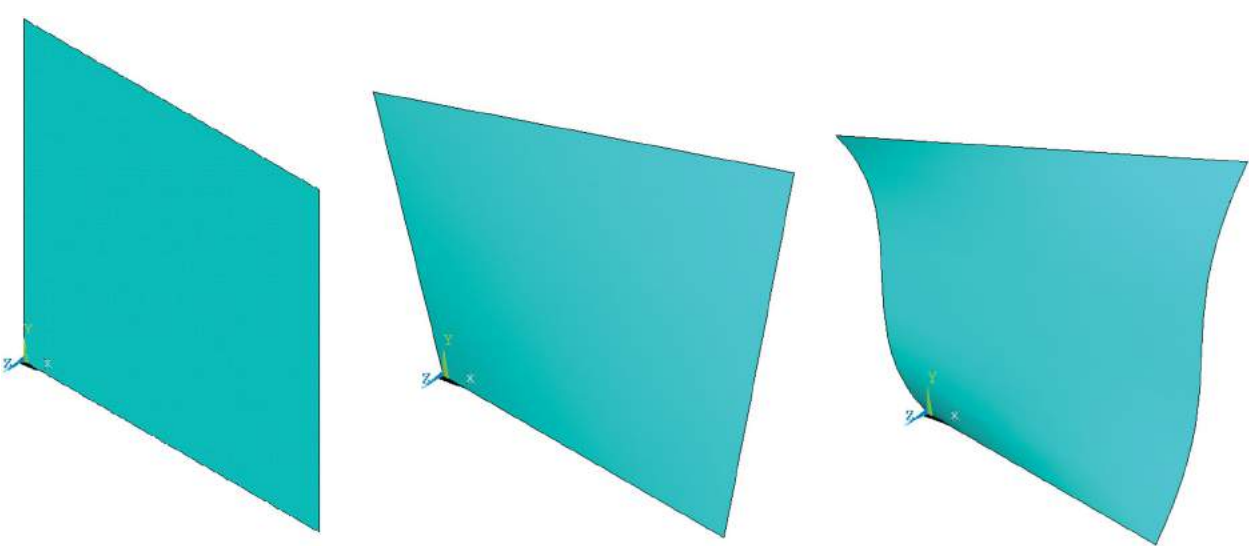

Figure 11. Isometric views of each test geometry.

Table 1. Confusion Matrix resulting from LOOCV of training data using k-NN

	Predicted Class								
Actual Class	Mode1	Mode2	Mode3	Mode4	Mode5	Mode6	Mode7	Mode8	Junk
Mode1	20	0	0	0	0	0	0	0	0
Mode2	0	20	0	0	0	0	0	0	0
Mode3	0	0	20	0	0	0	0	0	0
Mode4	0	0	0	20	0	0	0	0	0
Mode5	0	0	0	0	20	0	0	0	0
Mode6	0	0	0	0	0	20	0	0	0
Mode7	0	0	0	0	0	0	20	0	0
Mode8	0	0	0	0	0	0	0	20	0
Junk	0	0	0	0	1	0	0	0	39

As can be seen by the values in the diagonal elements of Table 1, the k-NN algorithm was able to learn the training data with 99.5% accuracy. Similar assessments were made with the other algorithms chosen. The algorithms were subjected to a paired t-test at a 5% significance level in Weka's Experimenter to assess their differences, and the results of which can be seen in Table 2. It is shown that there is no significant difference at the 5% significance level between the IBk algorithm and the RandomForest algorithm, whereas there is significant degradation in the J48 and SMO algorithms. Since it was found that there is not a significant difference between the IBk and Random Forest algorithms, IBk was used due to its simplicity, speed, and ability to learn the training set.

Table 2. Results of a Paired T-test

Dataset	(1)	(2)	(3)	(4)
Mode-Shape	99.50	99.50•	98.85	81.00•

○,• statistically significant improvement or degradation

Table 3. Parameters of baseline designs

	Parameters of Baseline Designs (*inches*)		
	Rectangular Plate	Tapered Twisted Linear Plate	Tapered Twisted Non-linear Plate
Length	20	20	20
Width	20	N/A	N/A
Top Width	N/A	20	20
Bottom Width	N/A	20	20
Thickness	0.5	0.05	0.05
Nodes along Width	25	25	25
Nodes along Length	25	25	25
Twist	N/A	1	1

5.2. Mode identification

By utilizing parametric geometries, vibrational mode shapes are easily identified without regards to mesh coarseness. Table 3 shows three geometrically identical models that have differing mesh densities and therefore a differing number of nodal displacements in their modal shape vectors. The results of testing this method include such variations of mesh densities. Mesh density is dictated by the number of nodes along the width and length of the model and are defined in the baseline designs reported in Table 3. Table 3 contains the parameters that define each model tested. In order for this method to be used in an iterative design process it must be able to identify modes that result from models that have varied some parameter value(s) of its baseline design. These changes in parameters may be small or large. Such change is illustrated in Fig. 12 where the length of the model on the right has increased from its baseline design on the left.

This method was subjected to three types of geometry as described earlier where each model's parameters were subjected to changes in their baseline design ranging from $\pm$ 10% to $\pm$ 100%. The DOE in this method was used to produce the varied geometry according to a defined parameter percent variation. The tests were performed by compiling test sets of various sizes ranging from 20 to 50 instances per test set. Tests were performed for each geometry type three times to produce an average accuracy. The average accuracies for the three generated tests are reported in Table 4. These results can be compared to the results from the research performed by Selin et al. shown in Table 5.

Throughout all comparable geometry and DOE parameter variation, the method employed in this paper showed improvements in identification accuracy. The results of testing the rectangular plate show significant improvement over previous research. It can be seen from Table 4 that the accuracy of the mode identification method developed in this paper is related to the amount of variation in model parameters. However, such degradation of accuracy does not come into effect until the variation reaches $\pm$ 60%. Throughout the full range of parameter variation the results show that the average accuracy never falls below 90% for the rectangular plate. Through $\pm$ 50% variation of the rectangular plate the accuracy was found to be 100%.

The tapered, twisted, linear plate showed a similar relationship between the amount of variation and the

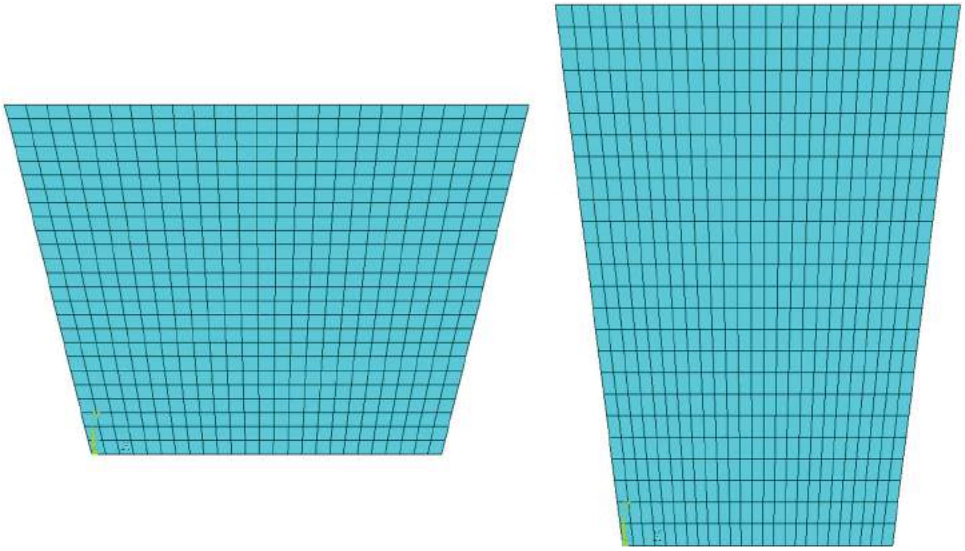

Figure 12. A baseline model and modified design.

Table 4. Accuracy of the method averaged over three tests

	Mode Identification Method Accuracy over 3 Tests									
	DOE Parameter Variation									
	± 10%	± 20%	± 30%	± 40%	± 50%	± 60%	± 70%	± 80%	± 90%	± 100%
Rectangular Plate	100%	100%	100%	100%	100%	99%	97%	97%	90%	98%
Tapered Twisted Linear Plate	100%	100%	100%	99%	96%	94%	92%	81%	76%	93%
Tapered Twisted Non-linear Plate	99%	100%	98%	97%	96%	97%	97%	89%	86%	84%

Table 5. Results from Selin et al. using surface templates

	Mode Identification Method Accuracy - Surfaces			
	DOE Parameter Variation			
	± 10%	± 20%	± 30%	± 40%
Rectangular Plate	96%	87%	82%	87%
Tapered Twisted Linear Plate	93%	90%	82%	84%
Tapered Twisted Non-linear Plate	91%	88%	87%	82%

method accuracy with amplified effects. This loss of accuracy is largely due to the fact that the baseline design of the tapered twisted linear plate has different parameters than those for the rectangular plate used in compiling the training data. By observing Table 4 it can be seen that this method was able to identify mode shapes with 100% accuracy through ± 30% variation. At ± 90% variation is found the lowest accuracy in the method of 76%. Next, the tapered twisted non-linear plate model was used for testing. This model, having the most differences from the model used in the training set, showed results consistent with the relationship between variation and the method accuracy. While staying above 95% accurate through ± 70% parameter variation, the average of the three tests show that for a tapered twisted non-linear plate this method is never 100% accurate. While the results show that at ± 20% variation the method has 100% accuracy, because it is only 99% accurate at the ± 10% variation level, we cannot say that this method is ever 100% accurate for the tapered twisted non-linear plate model. By investigating which mode shapes were incorrectly matched, it was determined that the mismatches were mainly caused by two problems. The first problem was when one mode closely resembled another and the distinctions between the two were difficult to identify even by visual inspection. This problem may be amplified when looking at higher order modes when distinguishing between mode shapes is subtle. The function used to create a NURBS surface from nodal displacements determines an approximate surface from a cloud of points. As it is an approximation and does not pass through each point, the surface may not represent the mode shape as accurately as possible. This approximation could add to the problem of misidentifying similar mode shapes. An example of how mode shapes can blend into one another can be seen in Fig. 13 where a distinct mode is represented at the ends of the figure and are conflated with each other towards the middle.

The second problem that was observed was when a mode shape had not been properly trained. For example in Figure 14 both contour plots represent the same mode in bending, but while examples of the mode on

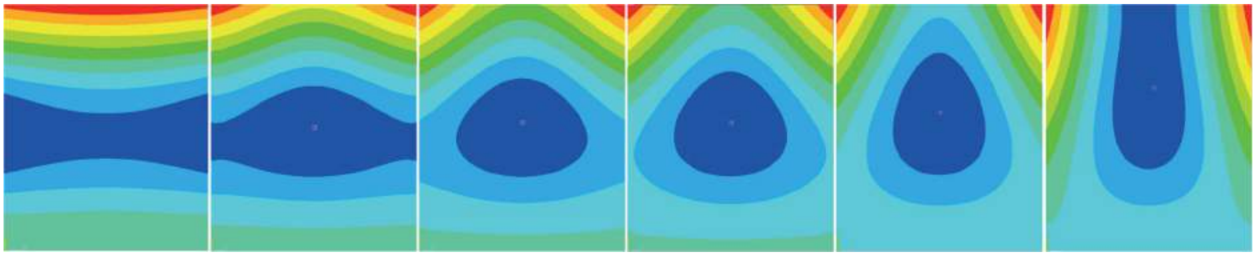

Figure 13. Sample of modes blending into another.

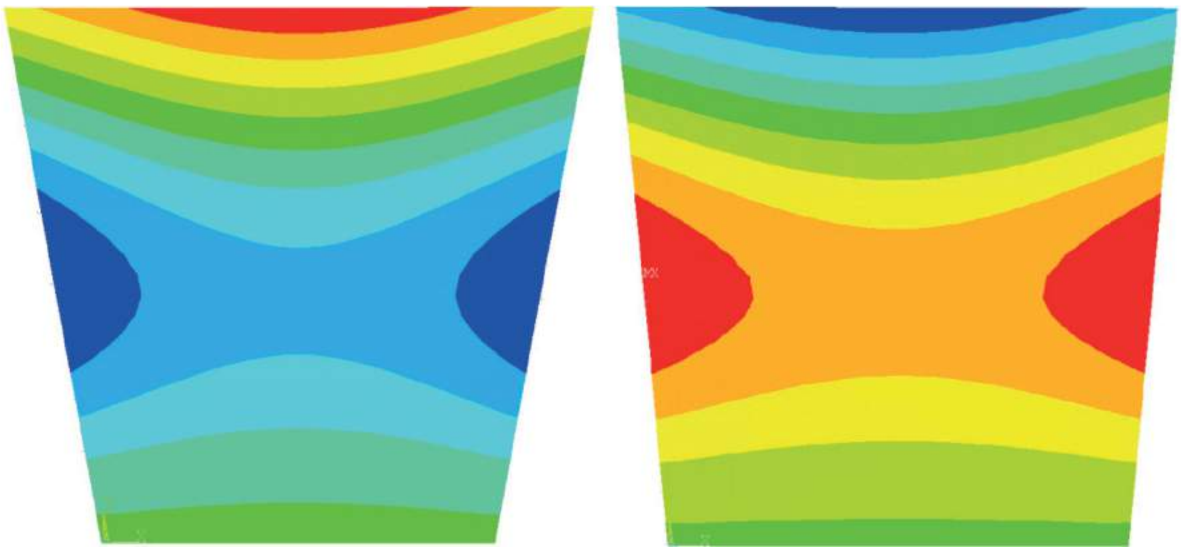

Figure 14. Sample of trained and untrained mode shape of same mode.

the left was included in the training set the mode on the right was not thus causing some error when the untrained mode was encountered. Both of these problems could be relieved by a larger training set that includes more examples of distinct mode shapes. The research performed has verified that by expanding the training set to include more examples these problems could be overcome. However, through continual use of this method new mode shapes would arise that exhibit the problems just mentioned and required further adjustment to the training set.

A final caveat that should be clear is that it was discussed that the displacements collected as attributes were in the z direction. This method looks at displacements perpendicular to a working plane. In this paper the working plane was the xy plane and as such z displacements were used as feature attributes. If the designed model is significantly displaced outside of the working plane information about the model could be lost. If in varying parameters of a design a model should be displaced in a direction that is not primarily perpendicular to the working plane, then the proposed methods effectiveness is not guaranteed and new attributes that can describe the model should be explored. For this reason a designed model should stay relatively in a working plane for best results. For established design processes this should not pose a problem as a working plane can be readily identified.

Despite the drawbacks of this method, the results show that the identification of mode shapes using machine learning and parametric surfaces is feasible. The results reported here demonstrate improvement in overall accuracy over a larger design space using machine learning over previous attempts using a MAC calculation. The advantage of using machine learning in this method is its ability to be modified to more accurately identify mode shapes. By modifying or adjusting algorithms or training data, a classifier can become more accurate in its predictions.

Machine learning also allows for additional descriptive attributes to be used in conjunction with such attributes of displacements used in this paper thus allowing for the identification of more complex parts. For example, by using the number of peaks and valleys present in a mode shape may help identify higher order modes.

5.3. Mode identification in iterative design

By automatically identifying mode shapes significant time savings have been realized over the current method of visual inspection. To test the time required to identify vibrational mode shapes in this method 40 runs were completed, each run producing 5 mode shapes to be identified. These results are benchmarked against the current method of visual inspection and to previous work performed by Selin et al. in Table 6. Due to the similarities in approach, the results for time required via visual inspection of mode shapes will be those reported by Selin et al. [16]. Selin's research used both parametric curves as well as surfaces. Since the proposed research uses parametric surfaces, the time saving reported herein will be comparing time to identify mode shapes using parametric surfaces only.

Table 6. Benchmarking time required of mode shape identification methods.

Identification Method	Total Time (sec)	# of Matches	Time per Match (sec)
Proposed Method	159.86	200	0.7993
Selin et al.	311.07	200	1.55535
Visual	2433.28	200	12.1664

Benchmarking the proposed method delivers time savings twice as fast as the method using the MAC and parametric surfaces proposed by Selin et al. and 15 times faster than visual inspection. While it is possible that using this method may result in an incorrect identification of mode shapes, the significant time savings that are achieved present a compelling argument for the use of this method even if the identification is not always 100% accurate.

6. Conclusions

This paper shows that machine learning can be effectively used to identify the mode shapes of dissimilar finite element models. This is possible through the representation of the model's modal analysis results as parametric surfaces which allows parts with different geometric definitions and mesh coarseness to be matched to trained mode shapes. By using parametric surfaces, the behavior of a mode shape can be well defined over the entire part. The nodal position and displacement data are used to create parameterized surfaces that represent specific mode shapes. Once the results from a finite element model are transferred into a parameterized geometric form, attributes are then easily collected to be used for classification using machine learning. Machine learning is then able to identify mode shapes between models of different mesh density and geometric definition given a trained classifier.

This method has shown to have high accuracy and over a large design space. The high accuracy achieved by this method suggests that the chosen attributes of displacements was sufficient in describing the mode shape surfaces used. The results also suggest that machine learning could be a valuable tool in identifying vibrational mode shapes of geometries other than four-sided surfaces provided proper feature attributes can be obtained. By automating the mode identification process, more detail about a part's dynamic properties can be obtained without having to visually inspect each modal solution. The greatest benefit of this paper can be realized when implementing an iterative design process, such as optimization or design of experiment. In iterative design processes the geometric parameters of a model are modified with each iteration. By using parametric surfaces to represent mode shapes, models of differing geometric or mesh properties to be successfully identified. While this method is not 100% accurate in the identification of the modes contained in the analysis results over all possible variations, its high accuracy can provide a designer with an understanding of the part's properties in the design process. Comparing the accuracies of this method with prior work by Selin et al. indicate that machine learning provides more accurate results over a broader design space. The time required to execute the proposed method has twice the time saving realized by Selin's work using parametric surfaces and 15 times faster than visual inspection of results. These time savings can become even more significant when a large number of modes are identified within an iterative process.

While this work was applied to two dimensional finite element models, the ability of machine learning to choose attributes that can describe a model's features can also be used to identify more complex models. For more complex analysis one would need to obtain feature attributes that can describe three dimensional finite element results. This could include number of sides or faces, location in Euclidian space after normalization, or some method of mapping a three dimensional object to u, v space in a similar manner as this paper employs. The main requirement to apply this method to more complex geometry is to identify some features to sufficiently describe the part in order to consistently identify its resulting mode shapes.

Acknowledgments

Special thanks to Pratt & Whitney for funding this work.

ORCID

Robert Porter http://orcid.org/0000-0002-4526-8807

Ammon Hepworth http://orcid.org/0000-0001-7651-1059
C. Greg Jensen http://orcid.org/0000-0003-1824-0945

References

[1] Alpaydin, E.: Introduction to Machine Learning, 3rd ed. MIT, Cambridge, Massachusetts, 2014.
[2] Alpha, W.: Paired t-test, http://mathworld.wolfram.com/Pairedt-Test.html, 2012.
[3] Bishop, C. M.: Pattern Recognition and Machine Learning, Springer, 2006.
[4] Burns, L.: MAC Evaluations Utilized in FEA Analysis for Mode Identification, Conference & Exposition on Structural Dynamics, 4(1), 2004, 2008–2017.
[5] Aha, D. W.; Kibler, D.; Albert, M. K.: Instance-based learning algorithms, Machine Learning, 6, 1991, 37–66.
[6] Ewins, D. J.: Modal Testing: Theory, Practice and Application, 2nd ed. Wiley, 2009.
[7] Xu, G.; Mourrain, B.; Duvigneau, R.; Galligo, A.: Parameterization of computational domain in isogeometric analysis: methods and comparison, Computer Methods in Applied Mechanics and Engineering, 200(23), 2011, 2021–2031. http://dx.doi.org/10.1016/j.cma.2011.03.005
[8] Gade, S.; Herlufsen, H.; Konstantin-hansen, H.: How to Determine the Modal Parameters of Simple Structures, Sound & Vibration, 36(1), 2002, 72–73.
[9] Paliouras, G.; Karkaletsis, V.; Spyropoulos, C. D.: Machine Learning and Its Applications: Advanced Lectures, Springer, 2001. http://dx.doi.org/10.1007/3-540-44673-7
[10] Harrington, P.: Machine Learning in Action, Manning, 2012.
[11] Hearn, G.; Testa, R.: Modal analysis for damage detection in structures, Journal of Structural Engineering, 170(10), 1991, 3042–3063. http://dx.doi.org/10.1061/(ASCE)0733-9445(1991)117:10(3042)
[12] Kotsiantis, S. B.: Supervised Machine Learning: A Review of Classification Techniques, Informatica, 31, 2007, 249–268.
[13] Hall, M.; Frank, E.; Holmes, G.; Phahringer, B.; Reuteman, P.; Witten, I. H: The weka data mining software: An update, SIGKDD Explorations, 11(1), 2009, 1–1. http://dx.doi.org/10.1145/1656274.1656278
[14] Mitchell, T. M.: Machine Learning, McGraw-Hill, 1997.
[15] Piegl, L.; Tiller, W.: The NURBS Book, Springer-Verlag Berlin Heidelberg, New York, NY, 1997. http://dx.doi.org/10.1007/978-3-642-59223-2
[16] Selin, E.: Application of parametric NURBS geometry to mode shape identification and the modal assurance criterion, Master's thesis, Brigham Young University, April 2012.
[17] Hughes, T. J. R.; Cottrell, J. A.; Bazileys, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Computer methods in applied mechanics and engineering, 194(39), 2004, 4135–4195.
[18] Witten, I. H.; Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed. Elsevier, 2005.
[19] Wolpert, D. H.; Macready, W. G.: No free lunch theorems for search, The Sante Fe Institute, February 1995.
[20] Zeid, I.: Mastering CAD/CAM, McGraw-Hill, Boston, 2005.