Object Classification from Analysis of Impact Acoustics

Robert S. Durst Eric P. Krotkov

Robotics Institute
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

We address the problem of autonomously classifying objects from the sounds they make when struck, and present results from different attempts to classify various items. We extract the two most significant spikes in the frequency domain as features, and show that accurate object classification based on these features is possible. Two techniques are discussed: a minimum-distance classifier and a hybrid minimum-distance/decision-tree classifier. Results from classifier trials show that object classification using the hybrid classifier can be done as accurately as using the minimum-distance classifier, but at lower computational expense.

1 Introduction

We[1] are interested in the problem of using the sound of impact to learn about the material composition of objects. As a first step towards this goal, we want to see if it is possible to accurately classify different objects based on the sounds they make when struck. Previous work in signal processing and pattern recognition has shown that acoustic-based classification of certain signals is feasible using either frequency-domain or time-domain models. We use spikes in the signal's frequency spectrum as feature vectors and show that classification is indeed possible.

One of this project's design paradigms is simplicity, so we demonstrate classification using a simple minimum-distance classifier. This scheme calculates the distance between a test vector and a prototype for every labeled class and assigns the test vector to the class whose prototype yields the smallest error. We use an obvious prototype definition: the representative vector for each class is simply the mean of the training samples' spectra. Experimental results suggest that using entire spectra as feature vectors may lead to poor classifier performance, so we develop a feature-extraction algorithm that identifies the N most significant frequency spikes and compiles the class means based on these extracted features.

In addition to the feature-extraction algorithm, we also develop the decision-map classifier, a variation of the standard minimum-distance classifier scheme. The decision-map classifier iteratively finds the closest prototypes to the test vector and assigns it to the nearest matching class. Preliminary results indicate that both the minimum-distance and decision-map classifiers can accurately classify objects based on the feature-extracted data, demonstrating that acoustic-based object classification is possible. Furthermore, we demonstrate by analysis that the decision-map classifier has a computational advantage over the minimum-distance classifier. The minimum-distance technique calculates the N-dimensional distance between vectors (where N is the number of features), while the decision-map calculates distance using only as many features as necessary to unambiguously find the closest class prototype. This can result in a significant reduction in computational complexity, making the decision-map classifier an attractive alternative.

2 Literature Review

Stewart and Larson [1] tried classifying aircraft based on acoustic emissions of significant stationary frequencies. The authors suggested that an appropriate classifier would be a frequency template-matching scheme.

Perron [2] classified feature-reduced signals using a decision tree instead of more analytical classifiers. She concluded that decision trees can perform reasonably well and offer a "natural" classification scheme; that is, one that is more conceptually meaningful than abstractions like linear discriminant functions.

This paper describes the decision-map classifier, a method that incorporates elements from the above techniques: feature extraction based on the N most significant frequency components and classification using a template matching scheme, implemented as a decision-directed minimum-distance classifier.

1. Current address for R. Durst: RL/ERC-1, Hanscom Air Force Base, MA 01731-3010.

0-8186-7108-4/95 $4.00 © 1995 IEEE

3 Test Setup and Design Goals

Data was taken by striking the object under test with a cane and recording the sound of impact. To strike the test objects, we used an aluminum cane with a plastic tip. To record sounds we used an omnidirectional electret condenser microphone, amplification circuitry, and a A/D board sitting in a 386-based PC. We found the hardware could sample up to 70kHz. Sampling triggered on the data's rising edge greater than 0.01V, and we prevented aliasing with a low-pass filter, which provided at least 30 dB attenuation at 20 kHz.

We tested five objects:

- standard clay brick, 19 x 8.9 x 5.7cm
- standard concrete brick, 19 x 8.9 x 5.7cm
- small ceramic tile, 11.4 x 11.4 x 0.6cm
- wooden block, 16.8 x 8.6 x 3.8cm
- irregularly-shaped zinc ingot, about 18 x 11 x 2.5cm

For each trial we dropped the cane from a fixed height onto the test object and sampled the sound of impact 4096 times at 40 kHz. We normalized this data to a peak value of 1 and subtracted the mean to remove the DC component. Next we tapered the data with a Hanning window, took the FFT and stored the power spectrum.

Upon triggering, the sampling hardware logged everything the microphone could detect, including environmental noise. We analyzed 16 samples of laboratory noise. Experimental results show that the noise spectrum has some definite and consistently-identifiable frequency components that could be identified as object features. Occasionally, background noise spikes were taller than some test objects' frequency spikes, so noise frequencies sometimes looked like object features. Typically, however, the test objects had different feature frequencies from the noise's features, so we could isolate the object's frequencies from the noise.

We also wanted to isolate the frequencies unique to the objects from the frequencies emitted by the cane. We found that, while intuition says that the sounds of two objects' striking are inseparable, the test objects remain identifiable even when struck with different canes.

We tried to predict what the sound of impact should be under ideal conditions by calculating the sound waves emitted by the object based on its composition and shape, and then verifying that the predicted values were close to the captured frequencies. Kac suggested that one could not infer the geometry of the membrane's bounded edge based on the sound of impact, unless this edge was a circle [3]. Fisher was more optimistic, and suggested that it was indeed possible to infer the shape of the membrane, but only if one knew the membrane's tropic structure [4].

As we did not know the molecular structure of the test objects, we experimentally determined what frequencies belonged to the objects by looking at *intra*-class comparisons using *different* canes: common features between trials using the same material and different canes imply representative object features. We tested four objects (clay and concrete bricks, tile and zinc block) using four canes (aluminum cane with a plastic tip, a wooden block, knuckles and fingernails).

Every object exhibited *frequency jitter*, small fluctuations in significant spikes' frequencies, across many trials using the same cane. The *best-behaved* objects, so called because they produced the most consistently-identifiable spectra, showed frequency jitter of two to three frequency bins (about 20 to 30 Hz). We discovered that most objects exhibited not only frequency jitter but also frequency *scatter*: the most significant frequency in one hit might be completely different from the main frequency in another hit for the same object. This suggests that the objects emit their most significant frequencies regardless of the cane used to strike them. Based on these results, we determined that "cane noise" would not hinder classification tests.

4 Extracted-Feature Classification

Since the frequency content of different materials appeared to be fairly distinct, we tried classification using a minimum-distance classifier, using objects' power spectra as test and training vectors.

4.1 Minimum-Distance Classifier Theory

In classical nearest-neighbor methods a test vector is compared with each labeled training sample and is assigned to the same class as the "closest matching" training vector, where "closest match" usually means the minimum distance measured in *feature space* onto which all vectors are projected.

If the number of labeled training samples is large, comparing every test vector with every training vector becomes impractical. In this case minimum-distance classification can be done by comparing test samples with prototypes for each class. An intuitive choice for a class prototype is the class mean (or *cluster mean*). Given a set of labeled training data, assume each vector $\mathbf{y}_j$ in class c_i bunches in the feature space with every other member of c_i. Then the arithmetic mean of the entire population of c_i is the cluster mean μ_i:

$$\mu_i = \left(\frac{1}{N_i}\right) \sum_{j=1}^{N_i} \mathbf{y}_{i_j} \qquad (1)$$

where $\mathbf{y}_{i_j}$ is the j^{th} labeled sample in class c_i and N_i is the number of vectors $\mathbf{y}_i$ in c_i.

Finally, error, or *distance,* is defined. Typically, distance between vectors is simply defined as the Euclidean distance, or

$$d_i^2 = (\mathbf{x} - \mu_i)^T (\mathbf{x} - \mu_i) \quad (2)$$

where $\mathbf{x}$ and μ_i are column vectors and d_i is a scalar.

In keeping with the design goal of simplicity, we tried minimum-distance classification using entire spectra as vectors and class means as prototypes. Overall performance was 45.9%, with a worst-case classification error of 100% for concrete. We assumed that the bad performance was due to inherent intra-class spectral variations, which we saw under controlled conditions. However, observation has shown that most materials' major spikes occupy definite frequency bands and that these bands are different for each material, so using the main spikes' frequency coordinates as features should be sufficient for classification in a template-matching scheme. Extracting the most significant frequencies yields parameters suitable for the classifier's input data.

4.2 Feature Extraction

Extracted-feature classifiers require that a prohibitively large data set be distilled into a meaningful but much smaller representation. Often we want to model a complex signal as a much simpler signal that can be described by unique parameters, or *features.*

References [2] and [5] showed that classifier performance depends not only on how the classifier is defined but also on what features are used. Therefore, one must ensure that appropriate features be used to model the data. Again, we chose features from the frequency domain. The simplest feature-extraction scheme was to take the N tallest *feature spikes* in the power spectrum and use their frequency coordinates as classifier inputs. It is important to note that "spike" and "feature" are not synonymous. We define a spike as any data point whose value is greater than both of its neighbors. Spike identification of this sort does not prevent "parasitic" spikes: those data points that are spikes in their own right on a microscopic scale but are also part of a macroscopic trend that defined a much larger spike (Figure 1). Thus we needed to smooth out these parasites, which were not *significant* spikes (that is, potential features).

The other problem was to distinguish between features and non-features. Originally we used a rectangular window to minimize frequency spread when taking the FFT. However, this created large sidelobes, and sometimes these sidelobes were much taller than other frequency components. Thus we required additional preprocessing to reject sidelobes. An immediate solution was to use a different window for the FFT—we selected a Hanning

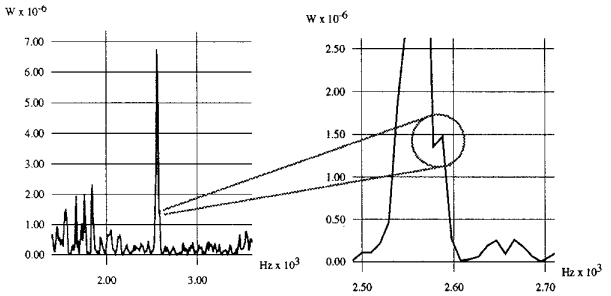


Figure 1. Parasite spike (circled) found on a more significant spike

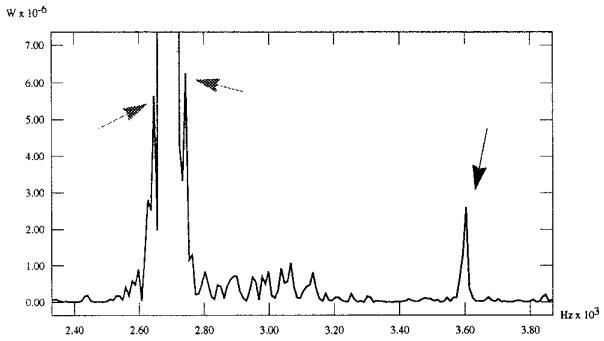


Figure 2. Close-up of a clay brick sample's power spectrum

window, which gave good sidelobe suppression. Still, for spikes whose amplitudes were orders of magnitude larger than other significant spikes (such as in the clay brick's spectra), even the suppressed sidelobes were sometimes larger than nearby feature spikes, as shown in Figure 2. Thus the question remained: how to discriminate a spike from another spike's remnant?

To reliably identify two equal-amplitude signals, they must be different in frequency by some minimum amount. This amount, the *resolvable frequency tolerance* f_{tol}, depends on the degree of separation. We used six frequency bins, or $f_{\text{tol}} = 58.6$ Hz, to provide 30 dB separation. That is, two signals of the same amplitude 58.6 Hz apart will produce a power spectrum that drops 30 dB from one spike before rising up to the next spike.

We assumed that every spike within f_{tol} from a larger spike was either a parasite or a sidelobe and rejected the smaller spike as a potential feature in favor of the larger one. The algorithm worked in a two-pass process. In the first pass it eliminated all non-spike data points. In the second pass, the algorithm took each spike's frequency f in turn and checked if there were any taller spikes in the interval $[f - f_{\text{tol}}, f + f_{\text{tol}}]$. If the current spike was tallest, it was stored in a new spike array. If there was a taller spike on this interval, we replaced the current spike with the taller spike. The process continued until all of the spikes either had been stored or erased. The spikes that remained were

the most significant spikes within their respective tolerance windows, and since we had eliminated all of the smaller surrounding spikes, the surviving spikes were guaranteed to separate by at least the minimum resolvable frequency tolerance. By definition, spikes that were separated by at least f_{tol} in frequency are features.

4.3 Prototype Definition

Because of the large size of each vector, we needed to generate a small set of prototypes for each class given the much larger set of training data. Because each test object exhibited frequency scatter, we could not use the *class* means as prototypes. Instead, we used *cluster* means of the extracted features. Even though the most significant extracted features from a given object's trials might not all be identical in frequency, we observed that usually the features naturally separated into several distinct frequency clusters per class, so we used *each* cluster's mean as *one of several* of that class' prototypes. Instead of comparing a test vector to each of five class prototypes, where each prototype was a single class mean, we had to compare the test vector to several prototypes for each class. In keeping with minimum-distance classifier theory, we assigned the test vector to the same class as the prototype that yielded the lowest error. Improving on minimum-distance theory, we use several clusters per feature per class, which allows for classification in the presence of frequency scatter. This section describes how we identified clusters and generated the prototypes.

Given: i classes with k samples per class. We extract N features per sample and label each sample's feature with ordinal *feature numbers*. A sample's "first feature" was the highest-amplitude spike in the power spectrum, the "second feature" was the second-tallest, and so on for all N features. We did this for all samples per class and looked at each class' extracted features by feature number.

For each class, we took each feature number n in turn and sorted all k n^{th}-feature spikes by frequency to identify each class' n^{th}-feature clusters. To define clusters, we assumed that two or more n^{th}-feature spikes that were close in frequency made a cluster, that clusters separated by at least f_{tol}, and that clusters spanned about $2f_{tol}$. We ignored as anomalies single features that did not belong to any clusters.

After identifying clusters, we calculated each cluster's arithmetic mean and variance, both in frequency and amplitude. These means and variances defined the prototypes. Note that each class may have several prototypes per feature number.

For wood, the features did not separate cleanly by at least the resolvable frequency tolerance f_{tol}. This required a more sophisticated clustering method than a single cursory review, so we chose the *k-means algorithm* [7]. We assumed there were five clusters; after six iterations the *k*-means algorithm stabilized and yielded usable feature groups.

4.4 Minimum-Distance Classification

We experimented with three minimum-distance classifiers on the feature-extracted data. Since we extracted only the first two features from the training and test data, the minimum-distance classifiers used two-dimensional distance measures, one dimension per feature. Each classifier differed only in its definition of distance, given in Equations 3 through 5. The distances d_i between a vector and class c_i's prototype are given for N features. f_j is the frequency coordinate of the test sample's j-th feature; F_{ij} is the frequency coordinate of c_i's prototype's j-th feature; x_j is the amplitude coordinate of the test sample's j-th feature; and X_{ij} is the amplitude coordinate of c_i's prototype's j-th feature.

a. $$d_i^2 = \sum_{j=1}^{N} \left(f_j - F_{i_j}\right)^2 \quad (3)$$

(sum of squared error; L^2-norm)

b. $$d_i = \sum_{j=1}^{N} \left|f_j - F_{i_j}\right| \quad (4)$$

(sum of absolute error; L^1-norm)

c. $$d_i^2 = \sum_{j=1}^{N} \left(\left(\frac{f_j - F_{i_j}}{\sigma_f} \right)^2 + \left(\frac{x_j - X_{i_j}}{\sigma_x} \right)^2 \right) \quad (5)$$

(modified Mahalanobis distance; removes bias due to spread in frequency and amplitude)

During program execution, the classifier read in a test sample and extracted the first two features to create a test vector, where each vector component was the test sample's ordinal features. That is, the test vector's first coordinate was the test sample's first feature and the test vector's second component was the test sample's second feature. Taking each class in turn, the classifier compared the prototypes to the test vector by feature number, and stored the nearest-neighbor prototype's feature as a "run-time" class prototype vector's component. Again, feature numbers corresponded to vector coordinates. This run-time class prototype came from a mixture of the closest features from all of the prototypes, and is the closest that class can match the test vector. Finally, the classifier compared the test vector to each of the run-time class vectors, using the same distance measure, to find the closest matching class. The results shown in Table 1 are for two-feature classification without rejecting any outliers. Judging by the

	distance formula	clay brick	concrete brick	ceramic tile	wood block	zinc ingot
Training Data	a	130/ 130	128/ 130	80/80	124/ 130	110/ 110
	b	130/ 130	128/ 130	80/80	122/ 130	110/ 110
	c	130/ 130	126/ 130	80/80	126/ 130	110/ 110
Test Data	a	37/40	38/40	17/20	31/40	98/100
	b	39/40	38/40	18/20	33/40	98/100
	c	36/40	35/40	17/20	36/40	92/100

Table 1: Extracted-Feature Minimum-Distance Classifier Results After Two Training Iterations

results, formal minimum-distance classification worked well using only two features, with an overall best-case accuracy of 94.2% on the test data after final training. It is plausible that performance could be improved slightly, using a parallel implementation of each distance measure. Taking the best-case classification results from the different distance formulas for each class yields three more correctly-classified test vectors (increasing performance to 95.4%). Depending on the application, this may or may not be worth the increase in classification time.

4.5 Decision-Map Classification

We also developed the *decision-map classifier*, a hybrid that combines elements from the minimum-distance classifier and a decision-tree classifier. Like a decision tree, the decision-map classifier refines its set of possible class assignments based on analysis of different features at different times. This analysis itself is a minimum-distance classification scheme: at each point, the decision-map classifier iteratively finds the nearest-neighbor prototypes in defined intervals around the test vector. The test sample is assigned to the closest matching prototype's class if and only if there are prototypes from a single class in the nearest occupied interval. If more than one class' prototypes are in the nearest occupied interval, the decision-map classifier moves to the test vector's next feature. This is analogous to moving down a decision tree.

During program execution, the classifier read in a test sample and extracted the first feature, then read each class' first-feature prototypes. Taking each prototype in turn, it defined an interval of radius $(\alpha\sigma_f + f_{\text{tol}})$ around the test sample's feature (initially, $\alpha = 3$), where σ_f is the standard deviation in frequency of the current prototype. If the prototype frequency was inside this interval, its class label

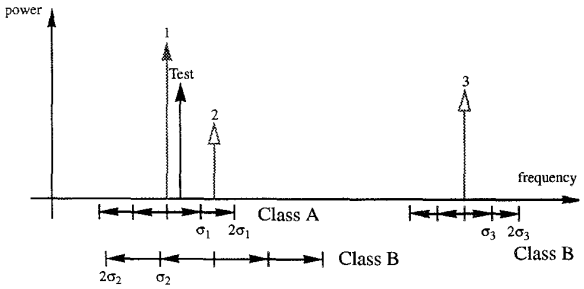


Figure 3. Example of decision-map classification using the first feature
Class A (denoted by solid gray arrows) has a single first-feature prototype and class B (hollow gray arrows) has two. The test vector (solid black arrow) is inside the $\alpha = 1$ interval to both class A's prototype (spike 1) and one of class B's prototypes (spike 2). Since the test vector is equidistant (in α) to both class' prototypes, classification is not possible using this feature.

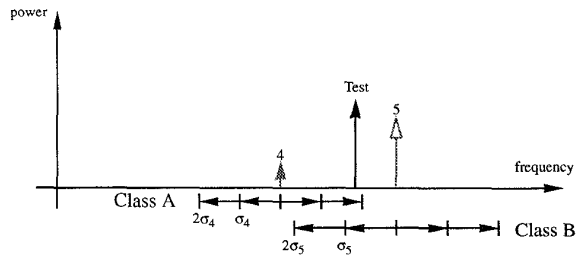


Figure 4. Decision-map classification using the second feature
As classification was not possible using the first feature in Figure 3, the decision-map uses the next feature. Class A is denoted by solid gray arrows and has one second-feature prototype; class B is denoted by hollow gray arrows and also has one second-feature prototype. The test vector's second feature (solid black arrow) is inside the $\alpha = 2$ interval to class A's prototype (spike 4) and is inside the $\alpha = 1$ interval to class B's prototype (spike 5). Since the test vector is closer in α to class B's prototype, the test sample is assigned to class B.

was stored along with the interval size α. The interval radius was shrunk by decrementing α and the routine repeated until $\alpha = 0$. This entire process was done for each class prototype for the current feature number.

The classifier produced its decision based on the stored α values. The stored class label associated with the smallest stored α was taken to be the winning class. If two or more different classes occupied the smallest α-level, no decision could be made for this feature number, and so the classifier would go to the next feature number and repeat the entire process until a winning class was found. Figures 3 and 4 illustrate the decision-map classification procedure.

Table 2 shows the decision-map classifier's results on test and training data. The decision-map classifier worked about as well as the minimum-distance classifier, with an overall accuracy of 93.8% on the extracted-feature test data. Even though it was implicitly based on a modified

Test Object	*Training Data*	*Test Data*
clay brick	130/130	39/40
concrete brick	128/130	37/40
ceramic tile	80/80	18/20
wood block	115/130	33/40
zinc ingot	110/110	98/100

Table 2: Decision-Map Classifier Results After Two Training Iterations

Mahalanobis distance measure, the decision map misclassified fewer samples than the formal Mahalanobis-based minimum-distance classifier did, performing about as well as the best formal minimum-distance classifier, which was based on the L^1-norm.

Certain details could be modified to potentially improve performance. One of these is the expansion of prototypes to higher feature numbers. Another reasonable improvement is to optimize the interval size around the current feature when identifying the prototypes closest to it. Currently, this interval takes on only three discrete values that depend on a variable datum for each prototype, namely, σ_f. If this variable were optimized for all prototypes based on the training data, there might be some slight improvement in classification. Both of these changes would occur *before* the classifier ran, so execution time compared to the nearest neighbor's would not change for the worse.

5 Summary

We are interested in the problem of classifying objects based on their acoustic emissions when physically struck. To this end we sampled five different objects: a clay brick, a concrete brick, a ceramic tile, a wooden block and a zinc ingot. Dropping a cane onto each, we hit the objects at various locations on their surfaces, and sampled and stored the resulting sound of impact. As previous work has shown that spectral elements of acoustic signals are sufficient for certain classification schemes, we used spikes in the frequency domain as classifier features.

A fundamental design goal of this project was simplicity, but one of the simplest pattern recognition techniques—a minimum-distance classifier that used entire spectra as input vectors—could not classify four of the five classes. We assumed this was due to joint effects of intra-class variation, large-valued spikes and the large size of the input vectors, so we developed a moving-maximum window to extract the N largest spikes in the frequency domain and used these spikes as feature vectors. After identifying clusters of each class' representative frequencies, we created class prototypes, vectors whose elements were the cluster means of the training data's extracted features. We then passed the prototypes and test samples' extracted-feature data to two kinds of classifiers, the minimum-distance classifier and the decision-map classifier.

The minimum-distance classifier calculated the error between a test vector and each prototype using every vector element (feature), and assigned the test sample to the prototype that yielded the lowest error measure. The decision-map classifier, a variation of the standard minimum-distance classifier, iteratively found the nearest prototype vector to the test sample based on each vector's most significant feature. If a single best match was found, the test vector was assigned to the same class. Otherwise, the classifier repeated the process using additional features as necessary.

While the decision-map classifier performed as well on feature-extracted data as the minimum-distance classifier, with 94% accuracy, it has a computational advantage. The minimum-distance classifier calculates its distance measure a fixed number of times that is a function of the number of features. The decision-map classifier, on the other hand, uses only as many features as necessary, so it can perform significantly fewer calculations than the minimum-distance classifier does.

6 Acknowledgments

The authors gratefully acknowledge the assistance of Henry Borysewicz.

7 References

[1] Stewart, C. and Larson, V. "Detection and Classification of Acoustic Signals from Fixed-Wing Aircraft," *IEEE Intl. Conf. Systems Engineering*, pp. 25-28, 1991.

[2] Perron, M.-C. "Feature Extraction and Learning Decision Rules from Ultrasonic Signals; Applicability in Non-Destructive Testing," *Proc. IEEE Ultrasonics Symposium*, vol. 1, pp. 533-6, 1988.

[3] Kac, M. "Can One Hear the Shape of a Drum?" *American Mathematical Monthly*, vol. 73, no 4, part II, pp. 1-23, 1966.

[4] Fisher, M. "On Hearing the Shape of a Drum," *Journal of Combinatorial Theory*, vol. 1, pp. 105-125, 1966.

[5] Beck, S. *et al.* "A Hybrid Neural Network Classifier of Short Duration Acoustic Signals," *Intl. Joint Conf. Neural Networks*, vol. 1, pp. 119-24, 1991.

[6] Oppenheim, A. and Schafer, R. *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[7] Duda, R. and Hart, P. *Pattern Classification and Scene Analysis*. New York: John Wiley and Sons, 1973.