SAE TECHNICAL PAPER SERIES

2004-01-1781

Crash Detection System Using Hidden Markov Models

Gautam B. Singh and Haiping Song
Department of Computer Science and Engineering, Oakland University

Clifford C. Chou
Passive Safety Research & Advanced Engineering, Ford Motor Company

Reprinted From: Safety Test Methodology, and Accelerated Testing and Vehicle Reliability (SP-1879)

ISBN 0 7680 1427-1

9 780768 014273

SAE International™

2004 SAE World Congress
Detroit, Michigan
March 8-11, 2004

400 Commonwealth Drive, Warrendale, PA 15096-0001 U.S.A. Tel: (724) 776-4841 Fax: (724) 776-5760 Web: www.sae.org

ISBN 0-7680-1319-4
Copyright © 2004 SAE International

Printed in USA

2004-01-1781

Crash Detection System Using Hidden Markov Models

Gautam B. Singh and Haiping Song
Department of Computer Science and Engineering, Oakland University

Clifford C. Chou
Passive Safety Research & Advanced Engineering, Ford Motor Company

ABSTRACT

This paper presents the design of a crash detection system based on the principles of continuous-mode Hidden Markov Models (HMM) with real-valued emission parameters. Our design utilizes log-likelihood for optimizing HMM parameters including the number of states in the model and the accelerometer crash-pulse buffer size resulting in lower costs and complexity of the crash detection system. Cross validation technique based on *Jackknifing* is utilized to estimate the crash pulse detection rate for a variety of crash events involving rigid as well as offset deformable barriers with head-on and oblique angle impacts. The system is simulated using Matlab and Simulink, and the proposed model is able to accurately classify crash-events within 10 ms from the time of the impact.

INTRODUCTION

With the vast majority of vehicle consumers citing safety feature as an important factor in purchasing decisions, crashworthiness of a vehicles plays an ever important role in establishing quality and customer acceptance. The new U.S. Government rulemaking for occupant protection using passive restraints also requires robust sensing systems to detect and deploy restraint systems to protect a wide range of occupants in a variety of crashes[1]. The automobile industry has come a long way in providing *smart* restraint systems that enhance the performance of the first-generation products[2]. An early detection of crash-event allows safety engineers to better tailor airbag inflation rates, pressures and belt pre-tensioning systems: thus sensing algorithms for timely airbag deployment are being constantly improved through the incorporation of increasingly complex processing logic. However, the current crash detection techniques are based on multi-stage sequential signal analysis algorithms and are generally tuned to detect specific types of crash events.

Existing crash detection algorithms can be divided into two categories – speed dependent and crush dependent. The speed-based sensors rely on variables related to speed, such as speed change (ΔV), jerk, speed, displacement, energy etc. The crush based sensors use the input from one or more sensors that are mounted in the crush zone and the compartment of the vehicle to predict the severity of the crash. So, in recent years, more complex algorithms are suggested, and machine learning and pattern recognition techniques have been applied to crash sensing. Barnard and Riesner developed an algorithm uses momentum and energy in semi-metric spaces[3]. Singh and Song developed an algorithm utilizing *discrete*-Hidden Markov Models to predict crashes[4]. Yin et al. suggested an algorithm based on artificial neural networks[5]. Krumm and Kirk use image-processing techniques to detect relative occupant motion within the vehicle compartment for airbag deployment[9].

Increased attention has been directed towards the side impacts and rollover sensing. In the case of side impact, the space between the interior and the occupant are much smaller than in the case of frontal impact. Therefore, the trigger times required to initiate the airbag inflation are very short, often less than 5 milliseconds for high-speed car-to-car impacts. Although rollover accidents occur less frequently than frontal or side impacts, they pose significant risk for serious occupant injury. It is estimated that in the United States, nearly one quarter of all fatal automotive accidents involve vehicle rollover. In both side impact and rollover cases, the need for sophisticated and robust algorithms is inevitable[11].

Hidden Markov Models are stochastic learning algorithms that have been successfully utilized in automatic speech recognition and control applications. Today, most state-of-the-art speech systems are based on HMM-models[10]. HMMs are extremely flexible and are particularly suitable for this application because there is no theoretical upper limit on the size of dataset utilized for training HMMs. That is, the over-training is not an issue with HMMs given that appropriate number of hidden states with enough observation distributions and sufficient training data is utilized for inducing the model.

HIDDEN MARKOV MODEL WITH REAL-VALUE EMISSIONS

In the real world, most of the observations are continuous signals instead of discrete symbols chosen from a finite alphabet. To apply discrete HMM to the real world problems, we first need to quantize the continuous signals. However, this introduces quantization errors and noise resulting in degradation of accuracy and often requires longer crash pulse examination before an accurate classification is possible. Therefore, it is advantageous to be able to use HMMs with continuous observation densities assuming that the observation itself is inherently a continuous signal. In order to use a continuous observation density, some restrictions have to be placed on the probability density function (PDF) to ensure that the parameters of the PDF can be re-estimated consistently[8].

We have tested that the crash pulses satisfy Gaussian distribution[4]. By using Gaussian density, the emissions can be easily represented by mean vector and covariance matrix. Compare to HMMs with discrete emissions, this model can have multiple observation spaces, which make the model more flexible and extensible.

HMMs with Gaussian emission are similar to HMMs with discrete emission except the emission probability $b_j(O)$ has been replaced by mean vector μ_j and covariance matrix Cov_j.

For multivariate Normal distribution, the emission probability can be written as

$$b_j(x) = \frac{1}{(2\pi)^{n/2}\sqrt{|Cov_j|}} \exp\left[-\frac{(x-\mu_j)'Cov_j^{-1}(x-\mu_j)}{2}\right] \quad (1)$$

HIDDEN MARKOV MODELS

The crash detection is a real-time process and thus a simpler model is the preferred to enable us in making crash classification within hard real-time deadlines. The fully-connected and left-right HMMs are the most commonly used models. Left-right HMM has only forward transitions implying that the state transitions may never occur that take the HMM back to a state that has left in the past. The left-right HMMs have been successfully used in many time series applications. The fully connected HMMs are not subject to any such restriction, and it is possible for the HMM to transition from any state to any other state during any time epoch. Figure-1 and Figure-2 show a 3-state fully-connected HMM and a left-right HMM respectively.

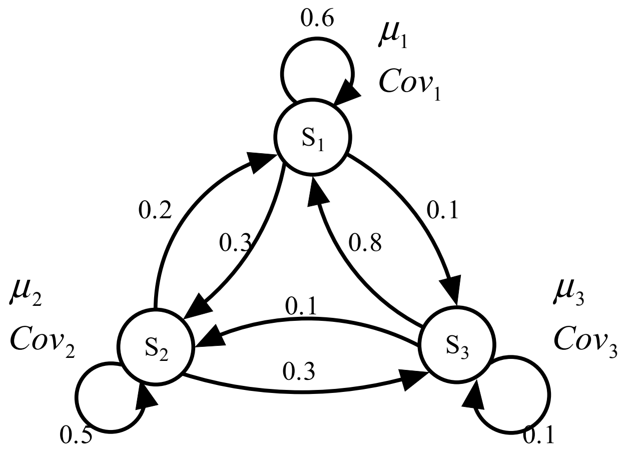


Figure-1: A 3-state fully-connected HMM

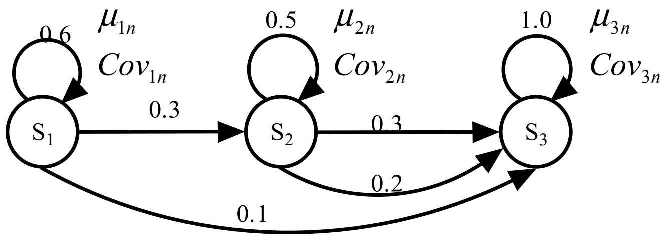


Figure-2: A 3-state left-right HMM

ESTIMATION OF THE NUMBER OF STATES OF A HIDDEN MARKOV MODEL, TOPOLOGY AND BUFFER SIZE

Since crash sensing is a real-time application simpler algorithms are preferred if the detection accuracy requirements are satisfied. In this study, we have developed experiments to optimize the HMM topology, the number of states, and the size of the buffer used for storing the past accelerometer readings. In order to choose suitable parameters for the HMM, we need a consistent estimator, based on the empirical crash pulses. Ziv and Merhav proposed an estimator of number of states of an HMM based on entropy and data compression[7]. Liu and Narayan also proposed an estimator by method of mixtures[6]. Here we use an estimator based on likelihood of crash pulses because likelihood is the by-product of HMM parameter estimation that is used to discriminate crash pulses from non-crash pulses. In statistical learning, likelihood can be written as $P(O \mid M)$, where *O* is the observation (data) under the model *M*. The likelihood is defined as the probability of the data (i.e. observation) given the underlying model. In our case, the likelihood ratio is calculated by considering the underlying models being a crash event model divided by the probability of observation under a non-crash event model.

To calculate the likelihood $P(O \mid M)$, we assume the observation sequence $O = O_1 O_2 \cdots O_T$, and the state sequence $Q = q_1 q_2 \cdots q_T$. The probability of the observation sequence O for the state sequence Q is

$$\begin{aligned} P(O \mid Q, M) &= \prod_{t=1}^{T} P(O_t \mid q_t, M) \\ &= b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T) \end{aligned} \tag{2}$$

The probability of such a state sequence Q can be written as:

$$P(Q \mid M) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T} \tag{3}$$

The probability of the O given the model is obtained by summing this joint probability over all possible state sequences *q* giving:

$$\begin{aligned} P(O \mid M) &= \sum_{\text{allQ}} P(O \mid Q, M) P(Q \mid M) \\ &= \sum \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \cdots a_{q_{T-1} q_T} b_{q_T}(O_T) \end{aligned} \tag{4}$$

To compute the likelihood efficiently, the forward variable $\alpha_t(i)$ is introduced. The forward variable is defined as

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i \mid M) \tag{5}$$

$\alpha_t(i)$ is the probability of the partial observation sequence, $O_1 O_2 \cdots O_t$ and state S_i at time *t*, given the model *M*. So the likelihood can be calculated by summarizing $\alpha_T(i)$ over all the possible states.

$$P(O \mid M) = \sum_{i=1}^{N} \alpha_T(i) \tag{6}$$

TRAINING HMM WITH REAL-VALUED EMISSION

Baum-Welch algorithm utilized for training. This is a special case of EM algorithm based on an iterative procedure, which chooses λ such that $P(O \mid \lambda)$ is locally maximized. It can be divided into E (Expectation) step and M (Maximization) step. In E-step, the likelihood $P(O \mid \lambda)$ is calculated. In M-step, we modify the model to maximize $P(O \mid \lambda)$ according to Baum's auxiliary function:

$$Q(\lambda, \bar{\lambda}) = \sum_{Q} P(Q \mid O, \lambda) \log\left[P(O, Q \mid \bar{\lambda})\right] \tag{7}$$

Baum and his colleagues have proved that maximization of $Q(\lambda, \bar{\lambda})$ leads to increased likelihood, i.e.

$$\max_{\bar{\lambda}} Q(\lambda, \bar{\lambda}) \Rightarrow P(O \mid \bar{\lambda}) \geq P(O \mid \lambda) \tag{8}$$

Baum-Welch algorithm for training a discrete HMM was previously described[4]. The training procedure is similar for HMMs with real-value emissions, except that in the M-step the mean vector μ_j and covariance matrix Cov_j need to be updated using the following equations.

$$\bar{\mu}_j = \frac{\sum_{t=1}^{T} \gamma_t(j) \cdot O_t}{\sum_{t=1}^{T} \gamma_t(j)} \tag{9}$$

$$\overline{Cov}_j = \frac{\sum_{t=1}^{T} \gamma_t(j) \cdot (O_t - \mu_j)(O_t - \mu_j)'}{\sum_{t=1}^{T} \gamma_t(j)} \tag{10}$$

JACK-KNIFING CROSS VALIDATION

When solving data mining problems the available data set is divided into training and evaluation sets. When there is insufficient data available for model training, cross-validation techniques such as "Jackknifing" provide statistically sound results using a minimal data set. In many cases, cross validation is the preferred method for obtaining statistically valid results. Cross validation is a heuristic that works as follow:

1. Divide the available data into a training set and an evaluation set.
2. Split the training data into *n* folds; each fold contains approximately the same number of crash pulses.
3. Set the parameters for the model being validated (in our case this is the HMM topology, number of states, accelerometer buffer size).
4. Use *(n-1)* folds to train a model, and the remaining one fold for testing.
5. Repeat steps 2 through 4 for each of the architectures – this is analogous to searching the architecture space for the optimal model.
6. Select the best model and train it using all the data from the training set.
7. Assess this final model using the evaluation set.

When *n* equals the number of patterns in the training data set, the procedure described above is also known as "Jack-knifing" cross validation.

In our case, we have 15 crash pulses. We use one third (5 pulses) as an evaluation data set, and the remaining 10 pulses as a training data set. The pulses chosen for training are shown in Figure-3. These 10 pulses are are further divided them into 10 folds, with each fold comprising of a 9:1 partitioning of unique training and testing pulses. While keeping the HMM topology fixed, a different set of training parameter settings (number of states, buffer size) is estimated. In the same manner, we also vary the HMM topology until the best possible crash detection results are achieved. Subsequently, after selecting the best topology and training parameters, all of the training data is utilized to train the final model, with the evaluation set being utilized for validating the accuracy of this final model.

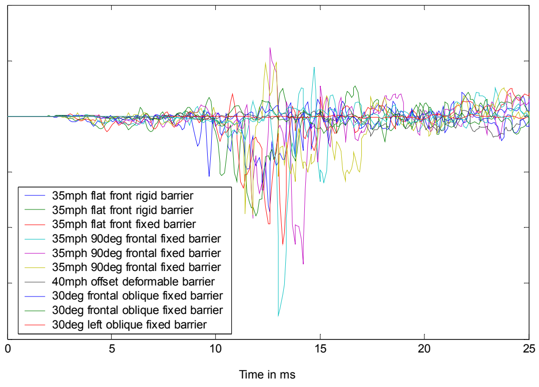
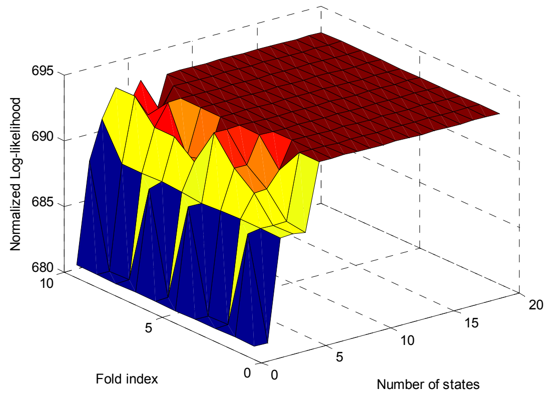


Figure-3. The ten training pulses used for training HMM parameters.

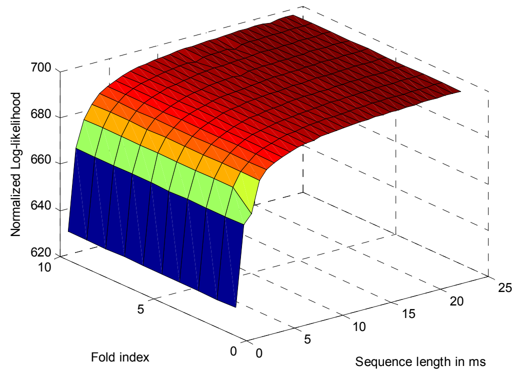
Figure-4 shows the *LL(N, F)* and *LL(L, F)* of the 10 pulses for the fully-connected HMMs, where *LL* is the normalized log-likelihood, *N* is the number of states of the HMM and *F* is the fold index, and *L* is the pulse length considered. In Figure-4, we can conclude that when $N \geq 8$ and $L \geq 5ms$, all of the folds are consistent fully-connected model.

Figure-5 shows the *LL(N, F)* and *LL(L, F)* of the 10 pulses for the left-right HMMs. In Figure-5, we can conclude that when $N \geq 7$ and $L \geq 5ms$, increase the number of states or buffer size will no longer improve the performance significantly.

In Figure-4 and Figure-5, given sufficient number of states and sequence length, all of the 10 pulses have a similar look of *LL(N,L)* no matter how different the original pulses is. This further validates our earlier observations that HMMs are noise tolerant and the estimator using Log-likelihood is consistent.

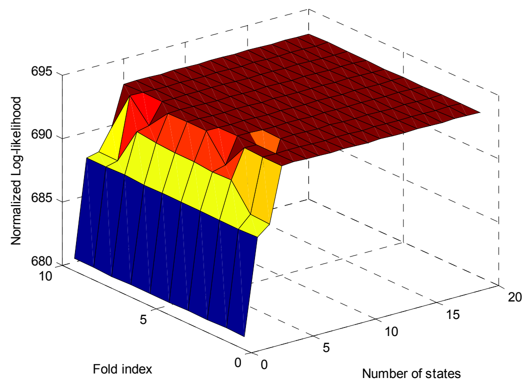


(a)

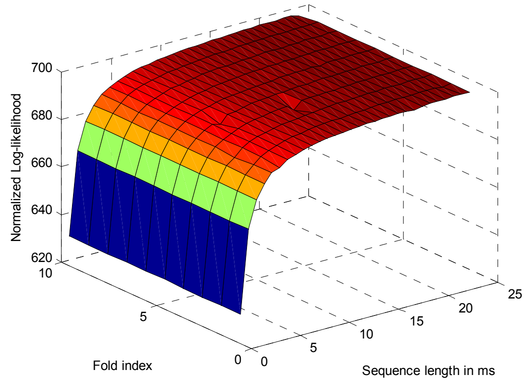


(b)

Figure-4 Jackknifing for the fully-connected HMMs. Z-axis represents the normalized log-likelihood of the 10 pulses, and Y-axis represents the fold index. In (a), X-axis represents the number of states from 1 to 20 . In (b), X-axis represents the sequence length in ms.



(a)



(b)

Figure-5 Jackknifing for the left-right HMMs. Z-axis represents the normalized log-likelihood of the 10 folds, and Y-axis represents the fold index. In (a), X-axis represents the number of states from 1 to 20. In (b), X-axis represents the sequence length in ms.

SIMULATION RESULTS

A left-right HMM with 7 states is trained using all the 10 crash pulses show in Figure-3, and the 5 pulses in evaluation set are used to evaluate the model. The buffer size for testing is 6 ms. The algorithm discussed above is implemented in Matlab/Simulink. We choose Matlab/Simulink because it provides a good platform for rapid prototyping and code generation for embedded systems. In Figure-6, the Judgment block generates a pulse when a crash is detected.

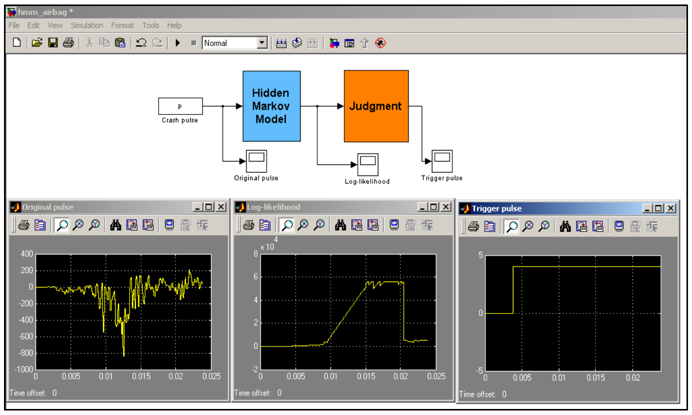


Figure-6 Simulating the HMM crash detection algorithm using Matlab/Simulink.

Figure-7 shows these 5 evaluation pulses, and Table-1 provides more information about these 5 pulses and the detection time by the HMM. From this table we conclude that *all the crash pulses can be detected within 10ms from the onset of impact.*

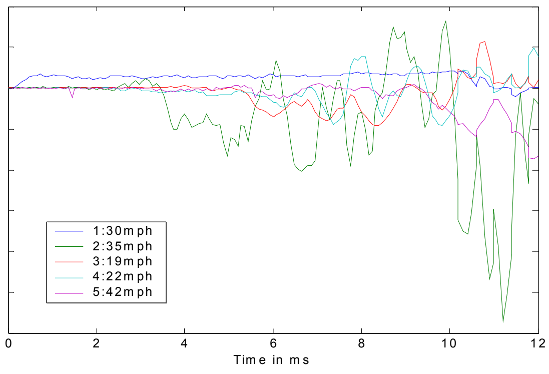


Figure-7. Evaluation pulses

Table-1 Detection time for the test pulses

Index	Speed (mph)	Type	Detection(ms)
1	30	90 deg frontal bumper override	5.52
2	35	Flat frontal fixed barrier	4.32
3	19	90 deg frontal center pole	7.20
4	22	30 deg frontal oblique fixed barrier	7.84
5	42	30 deg oblique front car to car	6.20

CONCLUSIONS

Using the Hidden Markov model methodology, this paper describes a modeling procedure for learning characteristics of crash pulses obtained from accelerometer measurements. The computational appeal of this modeling process is attributed to its learning based on the variability of signals which accounts for its performance in identifying new and unique type crash events. We tested the accuracy and speed of the algorithm, and established that a model when trained using a variety of crash signals exhibits capabilities in effective detection of the crash events in the evaluation data set used in this study within 10ms.

REFERENCES

1. Anson Foo and Setphen A.Ridella. "Advancements in crash sensing." **IEEE Vehicular Technology Society News,** November 2002: 9-13.
2. Ching-Yao Chan, On the detection of vehicular crashes – system characteristics and architecture, **IEEE Transactions on Vehicular Technology**, Vol. 51, No. 1, 2002
3. Robert Barnard and Miloslav Riesner, Vehicular air-bag control based on energy, momentum, and semi-metric spaces, **IEEE Transactions on Vehicular Technology**, Vol. 49, No. 5, 2000
4. Gautam Singh and Haiping Song, "Intelligent Algorithms for Early Detection of Automotive Crashes", **Society of Automotive Engineers**, *2002 World Congress & Exhibition,* Air Bags (Part A & B), Detroit, MI.
5. Yin Wuliang, Zhang Jinhuan and Huang Shilin, "Algorithm for airbag actuation based on artificial neural network", **Journal of Tsinghua University**, Vol. 39, No. 11, 1999.
6. C.-C. Liu and P. Narayan, "Order estimation and sequential universal data compression of a hidden markov source by the method of mixtures", **IEEE Transactions on Information Theory**, Vol. 40, pp. 1167-1180, 1994
7. J. Ziv and N. Merhav, "Estimating the number of states of a finite-state source**", IEEE Transactions on Information Theory**, Vol. 38, pp61-65, 1992
8. Lawrence R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition", **Proceedings of the IEEE**, Vol. 77, No. 2, 1989
9. John Krumm and Greg Kirk, "Video occupant detection for airbag deployment", **Fourth IEEE Workshop on Applications of Computer Vision**, October 1998, Princeton, New Jersey, USA
10. Jeff Bilmes, "What HMMs can do", **University of Washington Technical Report**, Number UWEETR-2002-0003, Jan. 2003
11. Automotive Crash Research: Side impact, rollover and vehicle aggressivity, Warrendale, PA : **Society of Automotive Engineers**, c2002