EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

A method for driving event detection using SAX with resource usage exploration on smartphone platform

Pimwadee Chaovalit*, Chalermpol Saiprasert and Thunyasit Pholprasit

Abstract

Driver errors such as careless and aggressive driving behaviors are one of the key factors contributing to road traffic accidents. It is, therefore, essential that drivers are aware of their actions when they are in control of the wheel responsible for not only their own lives but also passengers and bystanders on the road. Driver monitoring and advanced driver assistance systems have already been utilized in fleet and logistic domain as well as built into high-end vehicles commercially available in the market. However, the majority of drivers on the road today do not have access to such systems. This paper proposes a novel methodology of driving event detection using a time series approximation algorithm known as symbolic aggregate approximation (SAX) on data collected from smartphone sensors. The use of smartphone allows the system to be easily accessible, widely available, and implemented at low cost. In addition, a resource usage exploration on a smartphone platform is conducted in order to demonstrate the flexibility of our proposed algorithm to match different smartphone specifications. Preliminary results from our experiments revealed that the precision of the proposed detection algorithm of aggressive driving events is fairly good as the precision values range from 50% to 100%. In terms of resource usage exploration, it has been found that there is a strong linear relationship between the parameter settings for data compression and the runtime of the algorithm. This is beneficial when a trade-off is required between the accuracy of the algorithm and the resource usage on the smartphone.

Keywords: Driving event detection; Driving behavior; Smartphone; SAX

Introduction

Traveling from one place to another in a fastest possible time seems a necessity in our modern society today especially in big cities around the world. Drivers tend to be more aggressive and careless whether to change lanes quickly to avoid traffic or to overtake cars in front to beat the red lights. Consequently, this leads to an increased risk of road traffic accidents. Studies have shown that when a driver is monitored and driving events are recorded, the chances of aggressive and dangerous driving behavior are reduced [1]. An in-vehicle data recorder, similar to a black box on an aircraft, has been well established as a tool to record and store driving data occurring in a journey [2,3]. A number of commercial products available in the market using in-vehicle data recorders equipped with a wide variety of sensory devices such as Global Positioning System (GPS) receiver and often a video camera are used to monitor drivers [4]. Taxi operators and fleet management systems are some examples that deploy driver monitoring systems where every driver can be traced to ensure that they follow designated routes and do not violate the speed limit [5,6].

A rich source of data can also be obtained from real-time driving data through a network of sensory devices which can be used to detect possible collisions and crashes [7,8]. Car manufacturers have taken this idea and developed an advanced driver assistance systems (ADAS) such as collision prevention and avoidance systems [9]. This can only be found in high-end models at present as the sensors required for ADAS system are expensive making it very unlikely to be included in cheaper models.

*Correspondence: pimwadee.chaovalit@nectec.or.th
Information Communication and Computing Research Unit, National Electronics and Computer Technology Center, 112 Phahonyothin Road, Klong Neung Klong Luang, Pathumthani 12120, Thailand

It is well established that three main contributing factors to road traffic accidents are vehicle conditions, road infrastructures, and driver errors. In terms of driver errors, the fundamental elements leading up to aggressive driving behaviors are from different driving events that occur during a journey. These include sudden braking/acceleration, sudden turn, and sudden lane change. The sudden driving events can be distinguished from their non-sudden counterparts by a measurement of the acceleration rate per second. An acceleration is considered sudden when the acceleration rate is more than 0.3 G per second on a longitudinal axis. Similarly, a brake is sudden with the acceleration rate less than -0.3 G per second on a longitudinal axis. A right turn and a right lane change are sudden when the acceleration is measured on a lateral axis to be more than 0.3 G per second, and a left turn and a left lane change are sudden when the acceleration is measured to be less than -0.3 G per second. Hence, in order to recognize potential crashes, it is essential to be able to detect these fundamental driving events and classify whether or not they are aggressive.

So far, we have discussed about driver monitoring systems available and being deployed in the current market. These are based on in-vehicle data recorders which have the ability to store real-time driving data. The fact that these recorders are attached onto the vehicle means that they cannot be easily removed to be used in other vehicles. Hence, they are not flexible, impractical, and not very easily accessible. Modern smartphones available in the present market now come equipped with multi-sensors onboard which enable the capability similar to in-vehicle data recorder. In addition, they are easily accessible, widely available, and can be implemented at low cost. As a result, this makes smartphone a good candidate to be used as an alternative device to collect, process, and analyze driving data as well as detect and classify aggressive driving behaviors in order to alert drivers when they are being reckless.

Examples of the onboard smartphone sensors are accelerometer, GPS receiver, and gyroscope sensor. Accelerometer data provide an insight into the longitudinal and lateral movement of the phone while the onboard GPS receiver provides us with location data in terms of latitude and longitude. Therefore, these data allow us to infer the vehicle's movement when a smartphone is placed inside a car. The ability to detect these fundamental driving events would be highly advantageous to many application domains in the road safety perspective such as an automated advanced warning system. A number of studies have already deployed smartphones as a device to evaluate a driver's behavior and also road surface conditions in the literature [10,11]. However, the work proposed in this paper differs from the existing approaches in the literature as we deploy a technique called symbolic aggregate approximation (SAX) as an algorithm to detect driving events as opposed to dynamic time warping technique [11]. SAX offers the ability to efficiently approximate time series data which makes it a good candidate for implementation of real-time algorithm to be implemented on a smartphone. To the best of the authors' knowledge, this is the first time that SAX has been applied to driving data.

Another issue that is required to be considered is the computational power and resource usage on a smartphone platform as smartphones nowadays offer a wide range of hardware specifications with different levels of computing power and battery capacity. It is, therefore, highly beneficial that we are able to control how much resource is required when executing our proposed algorithm. One of the many features of SAX is the ability to vary parameter settings which enables us to control the resource usage for a given smartphone platform.

The paper is organized as follows. Related work is discussed in the section 'Related work' of this paper. Description of data collected from smartphone sensors is given in the section 'Description of sensors and data'. The methodology used in this paper is described in the section 'Methodology'. Experimental setup and results and analysis are presented in the 'Algorithm evaluation' and 'Exploration of smartphone resource usage' sections, respectively. Finally, the last section concludes this paper.

Related work

Driving behavior

Mohan et al. proposed a system where smartphones are utilized as a means to monitor road and traffic conditions [12]. This was achieved by using sensors onboard smartphones such as accelerometer and GPS sensor to detect potholes, bumps, as well as vehicles braking and honking. The system has been implemented and tested where promising results in terms of the effectiveness of sensing functions have been reported. Similar to [12], the approach proposed in [10] deploys a smartphone application which collects data from multi-sensors onboard to analyze road conditions obtaining at the same time high accuracy results in classifying different road defects.

Moreover, the work proposed in [10] presented a method to analyze a driver's behavior and advise drivers on sudden and harmful situation. Their approach is based on vehicle movements as well as physical road conditions, i.e., road surface roughness. However, their work mainly focused on the physical aspect of the road rather than the driver's behavior. The results from [10] revealed that their algorithm is able to classify different types of road surface imperfections such as bump and pothole with high accuracy.

Johnson and Trivedi proposed an approach in order to classify different driving styles based on data collected from smartphones [11]. In their approach, driving styles can be in the form of normal, aggressive, and very aggressive. A technique called dynamic time warping algorithm was used to detect driving events. This was achieved by matching incoming time series with the reference patterns of each driving event. The results from their work reveal that various sensors on smartphones can provide good source of information for an accurate measure and classification of different driving styles. Our previous work in [13] analyzed accelerometer and magnetometer data using pattern matching technique and also dynamic time warping algorithm to detect and classify aggressive driving events. Promising results have been reported with high percentage of precision and detection rate.

Our work proposed in this paper differs from the work in the literature as a novel algorithm based on the use of time series approximation technique called the *SAX*. SAX offers the ability to efficiently approximate time series data which makes it a good candidate for implementation of real-time algorithm to be implemented on a smartphone.

Approximation algorithm

In this paper, we explore the applicability of SAX on driving event detection. SAX is short for symbolic aggregate approximation. It is a technique which processes long time series into their approximation form. For driving event detection problem, we will have time series data whose values are collected data points from appropriate sensors onboard a smartphone. In addition, we will have pattern samples which represents driving patterns, i.e., sudden acceleration (SA), sudden brake (SB), and so on. By nature, pattern samples are short sections of time series validated by humans to be containing such driving patterns. We can view the driving event detection as a pattern matching problem between these pattern samples and driving episodes which are longer time series. We are interested in detecting driving patterns residing within driving episodes. We choose the SAX algorithm for various reasons.

First, SAX approximates time series into discrete levels of values. For example, a time series of 16 data points long may be approximated into three levels: low level, medium level, and high level. Those 16 sequential values can be analyzed by SAX and found that this time series start with high-level values, then low-level values, followed by middle-level values. The low-level, middle-level, and high-level values are designated as *a, b*, and *c*, respectively. Thus, our sample time series produce a SAX string of *caab*, as illustrated in Figure 1. The approximation technique utilized in SAX makes it a good candidate for real-time implementation on a smartphone as it requires less memory and execution time in comparison to raw time series.

Second, SAX helps to eliminate noise as it is an approximation technique. Therefore, SAX will be fit to use for noise reduction purpose with time series such as in the case of our data which is collected from sensors in a noisy environment, i.e., while driving on different road surfaces. Third, SAX comes ready with its own distance function 'min-dist' which is a lower boundary of the Euclidean distance [14]. For all the above reasons, SAX can be quite readily utilized onto our pattern matching problem of driving event detection.

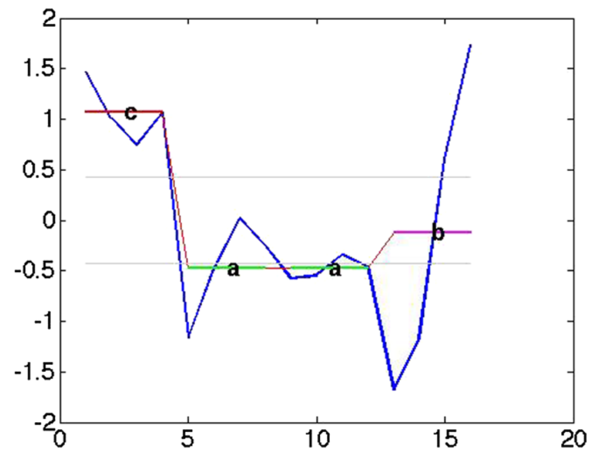

Figure 1 Discretized SAX transformed string.

Description of sensors and data

Three sensors from a smartphone are considered. Firstly, the three-axes accelerometer measures the force of acceleration whether caused by the phone's movement or gravity. The three axes correspond to lateral, longitudinal, and vertical accelerations. In this work, we are only interested in movements along the lateral and longitudinal axes which refers to side-to-side movement and forward and backward movement, respectively. In real-world scenarios, lateral acceleration or side-to-side movements represent driving events such as turning left and right and lane change while longitudinal acceleration corresponds to vehicle braking and accelerating.

The second sensor, magnetometer, measures the strength of magnetic field which can provide us a sense of direction at which the smartphone is pointing towards with respect to the magnetic north. It is a sensor usually found in a compass. Raw data from magnetometer will be utilized as an indicator for the detection of driving events in lateral domain. Our third sensor is a GPS receiver which provides positioning and speed data of the vehicle that the smartphone is attached to. Overall, accelerometer and magnetometer data are sampled at a rate of 5 Hz where one sample is recorded every 200 ms. Data from GPS receiver is sampled at 1 Hz.

Figure 2 shows a typical smartphone with relevant axes in accordance with the measurement from accelerometer.

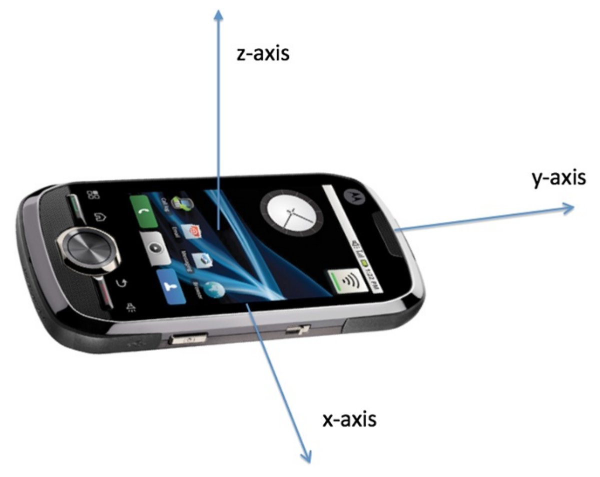

Figure 2 Three-axes acelerometer.

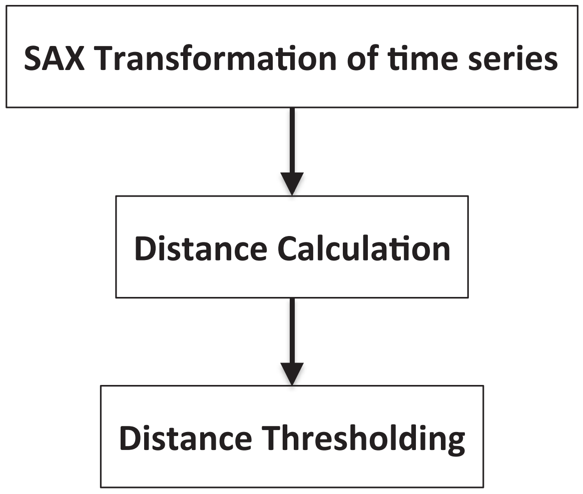

Figure 3 Overview of the proposed algorithm.

The lateral movement is denoted by the x-axis while the longitudinal movement is denoted by the y-axis.

Methodology

SAX is a time series approximation approach. It reduces a time series of length n to a string of length w, where $w < n$. w denotes the *word size* which is the length of the approximated time series while n is the original length of the time series. Within each string, there are a number of different alphabets taking place in a window of w. For example, a set of alphabets to represent a time series is $\{a, b, c, d, e, f\}$. Therefore, there are six different alphabets to represent this time series; *alphabet size* $(a) = 6$ in this case. First, a time series C of length n is reduced to a time series $\overline{C}$ of length w by the following equation [14]:

$$\bar{c}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} c_j, \qquad (1)$$

where $C = c_1, \ldots, c_n$ and $\overline{C} = \bar{c}_1, \ldots, \bar{c}_w$. From the equation, the algorithm first divides the original data equally into w frames. Second, it finds the mean value within the ith of w frame. The mean values have been normalized with zero as a mean of the transformed time series; thus, they follow the Gaussian distribution. Third, the mean values are discretized using breakpoints which divide the areas under Gaussian curve equally. The SAX algorithm was provided with a lookup statistical table. Then, each of the mean values in w frames can be mapped to the discretized symbols, producing a string of symbols referred to as *word*.

Figure 3 illustrates the overview of the proposed algorithm. It can be seen from the figure that our methodology of applying SAX comprises of three main steps described as follows.

Transformation of time series using SAX

For each pattern and driving episode to be considered a match, the first step requires that both pattern and driving episode data are transformed through the SAX algorithm. The transformation gives an approximated version of both pattern and driving episode, eliminating noise residing in the original patterns and/or driving episodes in the process. We utilized the MATLAB code of SAX downloaded from [15]. By calling the timeseries2symbol function in the MATLAB package, any input time series will be transformed to a string of SAX symbols. The function requires two pieces of information: word size *(w)* and alphabet size *(a)*. The first parameter, w, dictates the output length of the SAX string. The second parameter, a, dictates the number of ranges the values are discretized into. We may infer the *compression ratio* that SAX gives if we determine the original length of the input time series, where compression ratio $= w/n$.

In order to explore the applicability of SAX on driving event detection, we vary the word size parameter to four different compression ratios, that is 1/2, 1/3, 1/4, and 1/5. As for alphabet size, it was explored in [14] that the alphabet size may vary from the values of 2 to 8 and the best setting is data-dependent. We investigated the nature of driving pattern data by means of visualization. Figures 4, 5, 6, 7 illustrate sudden acceleration, sudden brake, sudden left turn, and sudden right turn driving patterns obtained from data collection. All of sudden acceleration patterns in Figure 4 show approximately the upside-down V or U shape of y-axis accelerometer, while all of sudden brake patterns in Figure 5 show approximately the V or U shape of y-axis accelerome-

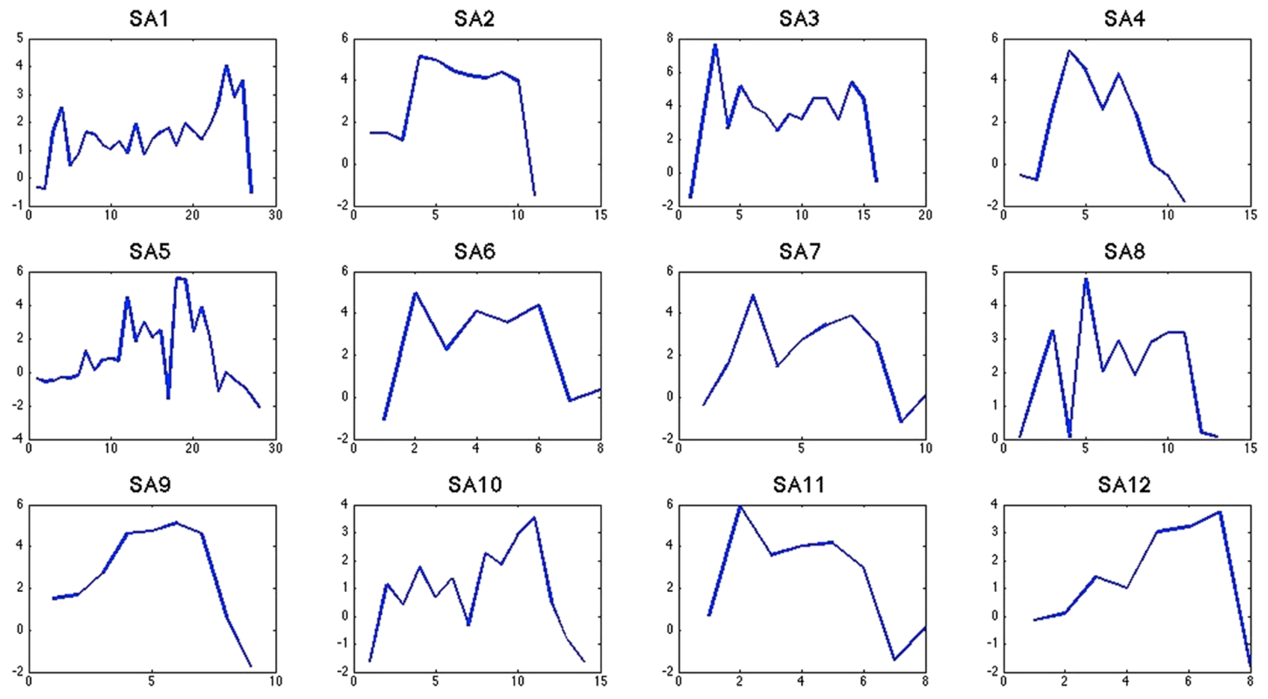

Figure 4 Sudden acceleration driving patterns.

ter. All of sudden left turn patterns in Figure 6 show approximately the V or U shape of x-axis accelerometer, while all of sudden right turn patterns in Figure 7 show approximately the upside-down V or U shape of x-axis accelerometer. From manually investigating the shape of all driving patterns, we found that most patterns have at least three levels of values. Hence, the minimum alphabet size was set to 3, and the maximum was set to the software limit of 10 in order to cover all data values.

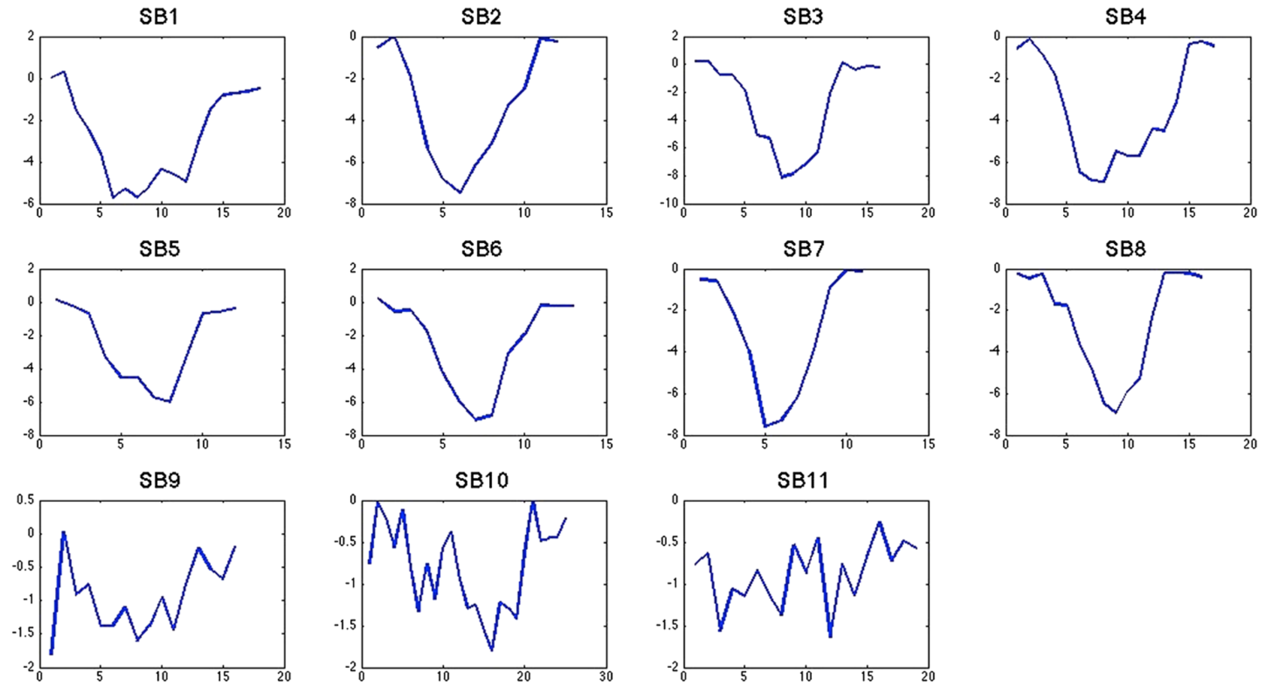

Figure 5 Sudden brake driving patterns.

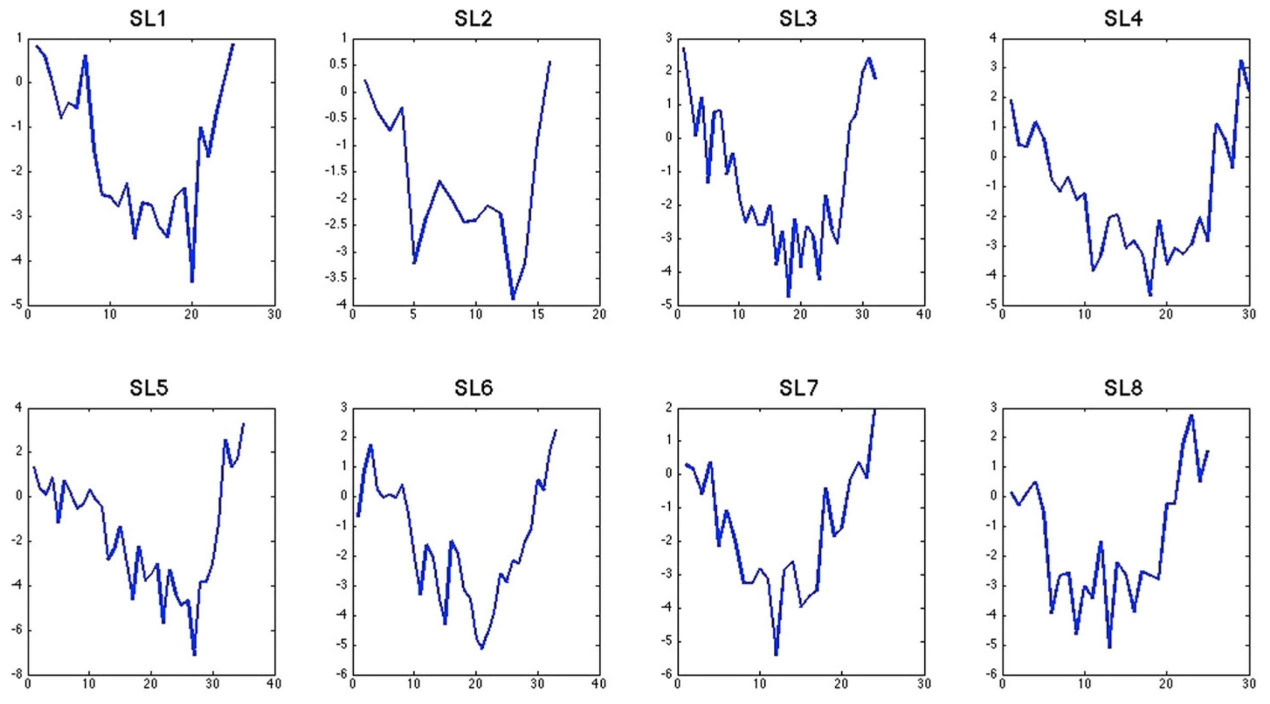

Figure 6 Sudden left turn driving patterns.

Distance calculation

After the transformation, both SAX strings of pattern and driving episode will be compared for their similarity to each other using a distance function. A distance function calculates how similar the two SAX strings are. When a distance function returns high values, two SAX strings are dissimilar. On the other hand, when a distance function returns low values, two SAX strings are alike. In general, there are various distance functions. For this study, we use the distance function which is suitable for comparing

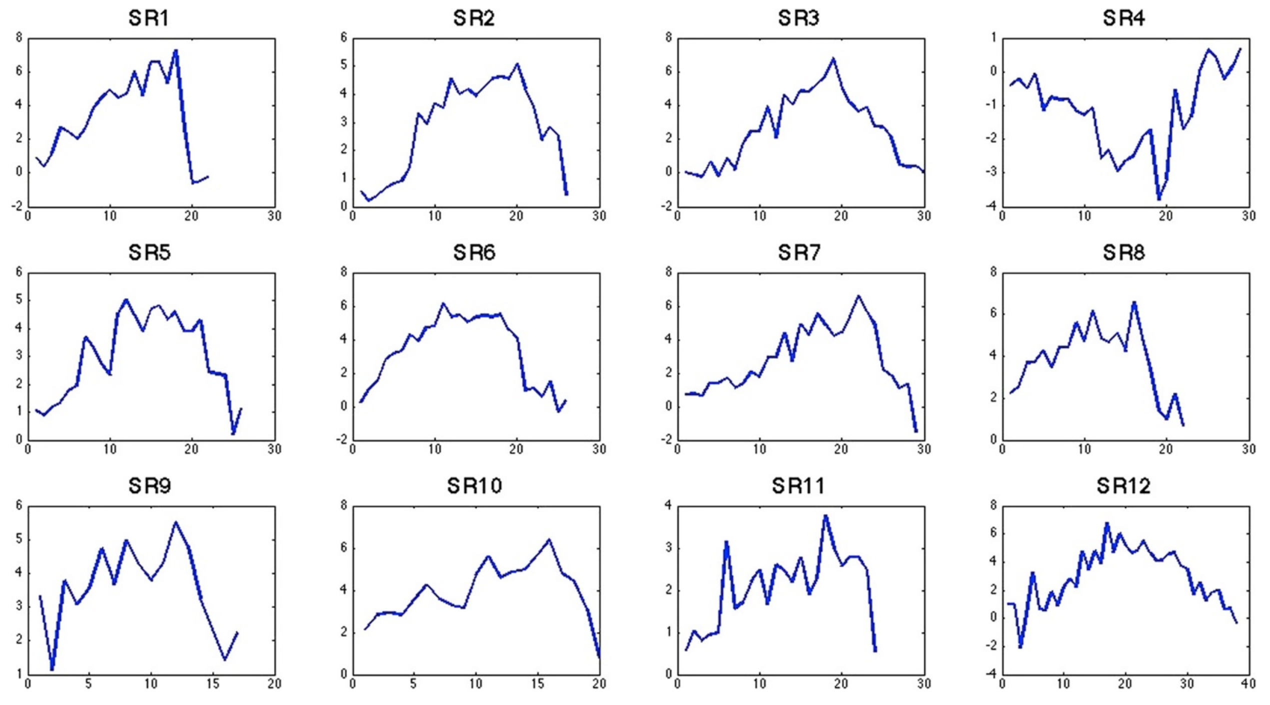

Figure 7 Sudden right turn driving patterns.

similarity between SAX strings, *min_dist* [14]. *min_dist* is expressed in (2), when the *dist*() function can be looked up from Table 1. Note that the lookup table (Table 1) varies according to the alphabet size *(a)*. Given two strings $\widehat{Q}$ and $\widehat{C}$, *min_dist* gives the root of sum square of distance values between the symbolic pairs from $\widehat{Q}$ and $\widehat{C}$, normalized by root inverse compression ratio. *min_dist* distance is proven to be a lower bound approximation of the Euclidean distance between original time series Q and C. Interested audience may refer to [14] for further implementation details.

$$MINDIST(\widehat{Q}\widehat{C}) \equiv \sqrt{\frac{n}{w}}\sqrt{\sum_{i=1}^{w}(dist(\hat{q}_i, \hat{c}_i))^2}, \qquad (2)$$

Distance thresholding

After calculating the distance between any two SAX strings, we now have distance values between the pair of strings. Distance values are numerical values whose ranges are dependent on the original values of the strings. However, distance values do not yet indicate how much alike two SAX strings are required to be in order to be considered a match. Therefore, a step is required to perform a thresholding of distance values in order to determine if and where the driving event occurs.

In our methodology, we experimented with a threshold variable t. The variable t defines the number of distance data points which gives the value of zero consecutively in order to detect the driving event. For instance, $t = 5$ dictates that the SAX strings of patterns and driving episodes are very similar where distance value equals 0 for 5 time periods. In a situation where a compression ratio is 1/5 and the original data sampling rate is 5 Hz, it has a temporal translation equivalent to that of 5 s.

Algorithm evaluation

In this experiment, raw data was collected using a single driver in one vehicle, 2010 Toyota Vigo, where approximately 120 driving events in urban and rural road environments were recorded. The route chosen is approximately 40 km long, from central Bangkok to the outskirt Bangkok, north west of the city. Data was collected using Android-based smartphone with a tailored made application which allows the tester to record driving events in real-time where a corresponding timestamp is noted in order to label our ground truth data.

Table 1 An example of *dist*() lookup table for using in *min_dist()* function

	a	b	c	d	e
a	0	0	0.59	1.09	1.68
b	0	0	0	0.5	1.09
c	0.59	0	0	0	0.59
d	1.09	0.5	0	0	0
e	1.68	1.09	0.59	0	0

The table is calculated for alphabet size (a) = 5.

Results and discussion

We run the SAX algorithm according to our methodology described in the previous section. We are interested in two main evaluation measures. First, we evaluate the precision of the driving event detection using SAX. Precision is defined as the accuracy of the detection. It represents the percentage that the algorithm correctly detects the event. Second, we evaluate the detection rate of SAX. Detection rate represents the percentage of observed events which are detected by the algorithm.

Figure 8 illustrates an instance of results obtained from using the proposed methodology in driving event detection, which in this case is SA. There are two sections displaying two time series. The time series on top is a section of driving episode, from which we are interested in detecting driving events. The time series on the bottom is a time-aligned section of distance values, which was calculated from the associated driving episode and a driving pattern of sudden acceleration.

From the ground truth, there are eight sessions of SA event between the following time point intervals: 5654-5670, 6237-6247, 6247-6262, 6262-6272, 6416-6443, 6551-6558, 6613-6622, and 6674-6686. From our method, the algorithm was able to detect three SA driving events at the following time points: 100, 6260, and 6553. Figure 8 presents both ground truths and detection points. Due to page limitation, only the area of plots from time point 5600 to time point 6880 was displayed. All eight ground-truth SA sessions are presented in the figure using opaque colored boxes overlaying the plot of driving event. For a setting of compression ratio = 1/2, alphabet size (a) = 4, and threshold (t) = 4, the algorithm detected three SA driving events as previously mentioned.

The first detection at time point 100 was out of range in the plot and was not the right detection. However, the other two detections at time points 6260 and 6553 were able to detect SA driving events accurately. The detection was presented using red arrows. As we can see, the red arrows point right at the correct time frame of the ground truth. As part of the evaluation, we allow a lag time of 3 s for the detection later than the driving event for it to be correct. This decision is based on an assumption that the algorithm needs to gather information on the fly. For a driving pattern to be detected, all or almost all of the pattern needs to be seen by the algorithm before the algorithm can make a decision. However, the

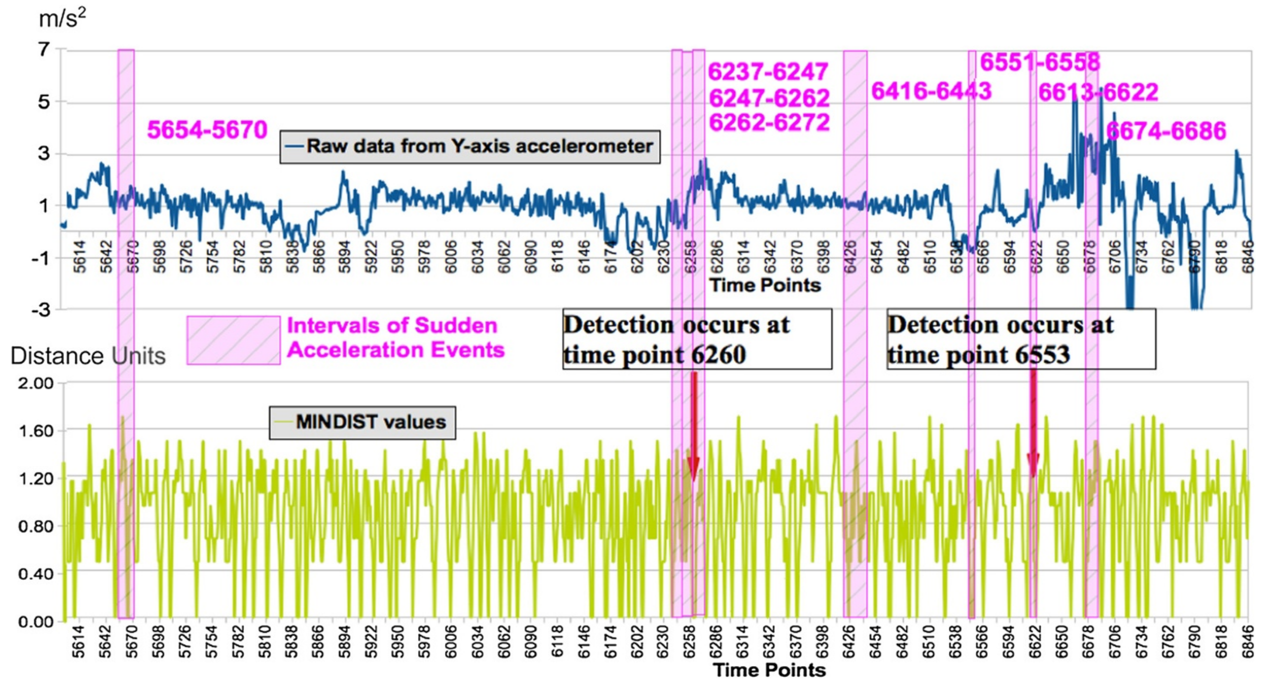

Figure 8 Sudden acceleration driving events analysis.

algorithm cannot take too long to produce the decision, otherwise the feedback of dangerous driving behavior will not be given in time. Therefore, the algorithm detected three instances of the SA driving pattern, two of which are correct. Moreover, from all of eight ground-truth sessions, three of which are detected. The detection at time point 6260 was counted as a detection for two sessions, 6237-6247 and 6247-6262. This gives the precision value of 66.67% and detection rate of 37.5% as presented in Table 2.

Results of all driving pattern detections are presented in Table 2.

From Table 2, each pattern has been detected with the SAX algorithm and the number of detected events is presented in the second column. In the third column, precision values of driving event detection range from 50% to 66.67%, with sudden acceleration and sudden left turn pattern detection having the best accuracy. The parentheses under precision percentages within each precision cell are the number of correctly detected events and the number of detected events. For example, three events were detected for pattern SA, two of which were correctly ground truth events. Even though precision values are mostly 60% and above (with the exception of sudden brake and lane change right pattern), detection rates obtained from the experiment (in the fourth column) are quite low, ranging from 20% to 37.5%. The parentheses under detection rate percentages within each cell are the number of real driving events detected by the algorithm

Table 2 Precision and detection rate in detecting 11 driving events (SA, SB, SL, SR, A, B, L, R, CL, CR, and U)

Driving event	Number of detected events	Precision	Detection rate
Sudden acceleration	3	66.67%	37.5%
(SA)		(2 of 3)	(3 of 8)
Sudden brake	4	50%	25%
(SB)		(2 of 4)	(2 of 8)
Sudden left turn	3	66.67%	25%
(SL)		(2 of 3)	(2 of 8)
Sudden right turn	5	60%	27.27%
(SR)		(3 of 5)	(3 of 11)
Acceleration	3	66.67%	50%
(A)		(2 of 3)	(2 of 4)
Brake	1	100%	20%
(B)		(1 of 1)	(1 of 5)
Left turn	1	100%	16.67%
(L)		(1 of 1)	(1 of 6)
Right turn	4	100%	20%
(R)		(4 of 4)	(1 of 5)
Lane change left	1	100%	16.67%
(CL)		(1 of 1)	(1 of 6)
Lane change right	1	50%	100%
(CR)		(1 of 2)	(1 of 1)
U-turn	7	71.43%	100%
(U)		(5 of 7)	(1 of 1)

and the number of ground truth events. For example, eight SA events actually occurred during the test, three of which were detected by the algorithm. This low detection rate can be improved with different parameter settings in our proposed algorithm which we will take note for future work.

Exploration of smartphone resource usage

Due to the nature of SAX, our algorithm has the ability to fine-tune the parameter setting which governs how the driving patterns and episodes are to be approximated. For example, alphabet size represents the discretized ranges of values that our raw data can be approximated into while the compression ratio, which is the ratio of the length of the input signal over that of the output signal, determines how much we would like to approximate the data by. Hence, by adjusting the compression ratio or the alphabet size, we are able to control the computational cost when the algorithm is in operation. This feature is very much suited for deployment on smartphone platform as smartphones are available in a wide range of hardware specifications where lower-end smartphones will possess less computational power and battery resources in comparison to high-end models.

A series of experiments were set up in order to demonstrate and evaluate the flexibility of our proposed algorithm in terms of resource usage on smartphone platform. More specifically, we examine computational time, CPU usage, memory usage, and battery usage of our proposed algorithm when it is in operation.

Experimental setup

Our proposed algorithm was implemented on HTC One X smartphone running Android OS 4.1. The phone is equipped with GPS receiver and accelerometer sensor which are the prerequisites for our proposed approach. Similar settings to the previous section was applied in this experiment with the sampling rate of accelerometer set to 5 Hz while GPS data is sampled at 1 Hz. We randomly selected a portion of our driving data for our data set to be used as input for this experiment. The two governing parameters, the alphabet size and the compression ratio, are varied to create a range of algorithm settings. Note that the same data set is used throughout this experiment for all algorithm settings.

A tailored made application was created with our proposed algorithm on board. With this application, we are able to obtain the execution time of the algorithm, CPU usage, memory usage, and battery usage. The definition of each parameter is as follows. Execution time is measured per iteration of the algorithm which is the time taken to process one new data point. CPU usage is the percentage of capacity of CPU required to perform one iteration of the algorithm while memory usage is the amount of memory required for the same task. In this work, battery usage is measured in mA for 100 iterations of algorithm execution. To ensure that other processes and applications have no effect on the obtained resource usage values, the memory is cleared before each execution and no other applications is initiated during the experiment. We have set the length of our driving episode to be 1,000 data points, $n = 1,000$, with our driving event pattern set to have 66 data points, $w = 66$.

Results and discussion

In total, 1,000 data points from driving episode were used in this experiment while we selected one driving pattern to be compared with the given driving episode. For a real-time environment, measurements of resource usage that we are interested in were taken after every one new data point has been sampled.

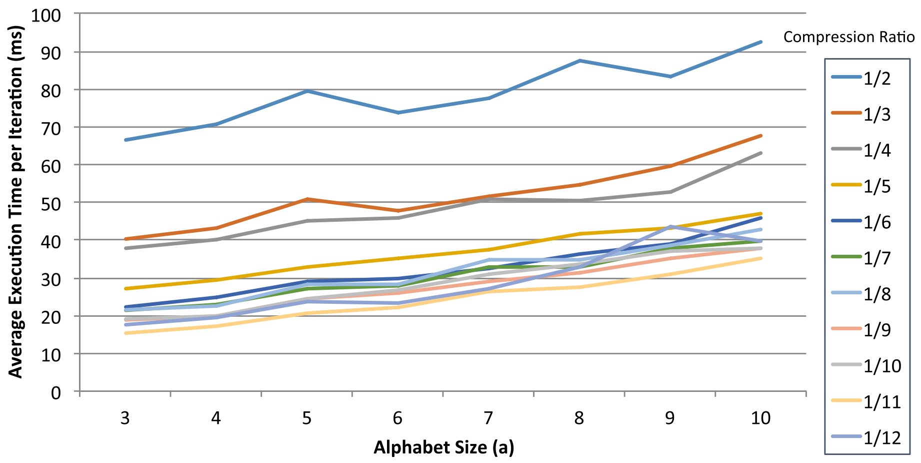

Figure 9 Plot of average execution time per iteration for a range of compression ratio and alphabet size settings.

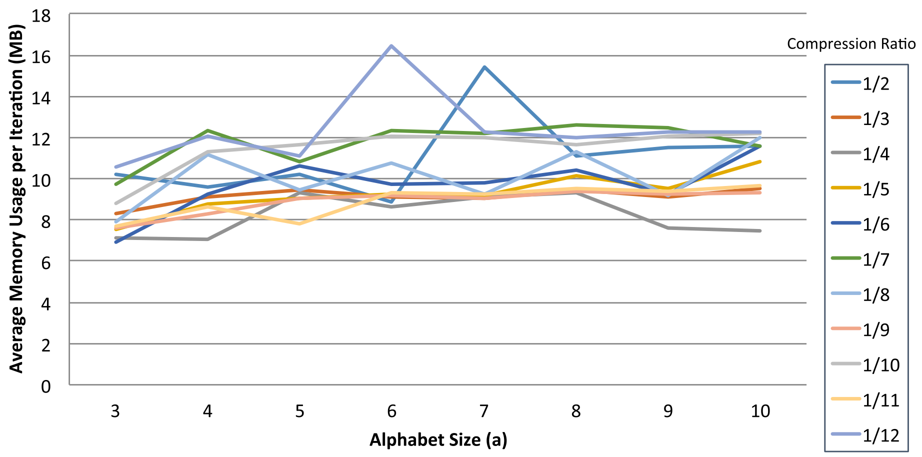

Figure 10 Plot of average memory usage per iteration for a range of compression ratio and alphabet size settings.

Figure 9 illustrates a plot of average execution time per iteration of the algorithm to process a window of one new data point. The execution time includes the three main steps of our proposed algorithm which are transforming a selected portion of the driving episode to a SAX string and then performing a distance calculation with the driving pattern SAX string followed by distance thresholding. The plot shows a strong linear relationship between the average execution time per iteration, the compression ratio, and the alphabet size. From the plot, it can be seen clearly that when the compression ratio is set to 1/2, our algorithm takes the most amount of time to complete one iteration for all ranges of alphabet size as the line is very distinct and well separated from the other compression ratio values. As expected, the average execution time per iteration decreases as the compression ratio decreases since we will have less data points to process after SAX transformation.

Another important point to note from Figure 9 is that the effect of compressing more data becomes less significant on the execution time below the compression ratio of 1/5 as the plots seem to saturate at approximately 15 to 35 ms. This is significant because increasing the compression ratio will likely decrease the accuracy of the driving event detection algorithm.

Figures 10 and 11 illustrate the plots of average memory usage per iteration and the battery usage per 100 iteration, respectively. It can be seen from both plots that there is no apparent relationship between compression ratio, alphabet size, and either of the memory or battery usage. The data on the graphs are clustered into one area with one or two outliers. One possible reason for these outliers is that Android operating system has other processes which we have no control of running in the background to optimize memory usage.

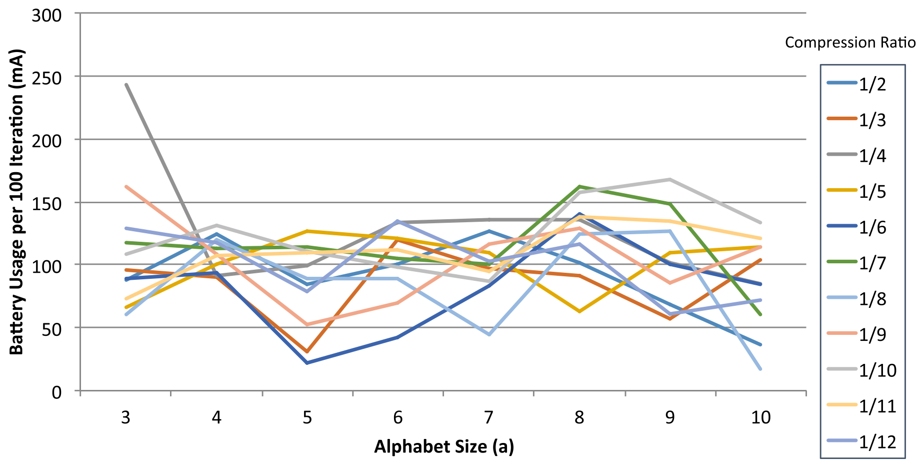

Figure 11 Plotzof battery usage per 100 iteration for a range of compression ratio and alphabet size settings.

For the memory usage, we can conclude that our proposed algorithm requires approximately 7 to 13 MB of RAM to process one iteration for this given data set. This suggests that our algorithm is suitable for implementation on lower-end smartphones that contain either 256 or 512 MB of RAM. On the other hand, approximately 50 to 160 mA of power is required to execute 100 iterations of our algorithm. The battery on our test device is quoted to be 1,800 mAh. Hence, running our algorithm alone on this test device will last approximately 11 h.

The readings of CPU usage have also been taken. However, they were found to be very small ranging between 0% and 1%. Therefore, it can be concluded that our proposed algorithm is not computationally expensive as less than 1% of CPU usage is reported for all ranges of parameter settings for this given data set.

Conclusions

This paper proposes a novel method of applying a time series approximation technique called the SAX on driving event detection problems. The proposed system utilizes a modern smartphone as a means to collect raw driving data based on the onboard sensors picking up the car's maneuvers when placed inside a vehicle. The use of smartphone allows the system to be easily accessible, widely available, and implemented at low cost. Real-world data is collected through an experiment conducted on a route containing both urban and rural roadways. Preliminary results from our experiments revealed that the precision of the proposed detection algorithm of aggressive driving events is fairly good as the precision values range from 50% to 100%. From the plot in Figure 8, the algorithm can detect driving events which are hard to detect with bare eyes and does so in a timely fashion. A resource usage exploration has also been conducted on a smartphone platform to demonstrate the flexibility of our proposed algorithm. It has been found that there is a strong linear relationship between the parameter settings for data compression and the runtime of the algorithm. This is beneficial when we want to trade-off the accuracy of the algorithm and the resource usage on the smartphone.

Further improvements can be made as our future work on the detection rate of the proposed algorithm. We believe that the low detection rate can be improved from a better understanding and optimization of parameter settings for the approximation of driving patterns and episodes. We hope in the future to execute more repeated tests on a greater number of events in order to provide sufficient test instances of some statistical significance.

Competing interests
The authors declare having applied for a patent related to the content of this research article.

Acknowledgements
This research is jointly funded by National Electronics and Computer Technology Centre (NECTEC) and Asian Transportation Research Society (ATRANS) grant number 12-13/001.

Received: 31 January 2014 Accepted: 14 July 2014
Published: 18 August 2014

References

1. JS Hickman, ES Geller, Self-management to increase safe driving among short-haul truck drivers. J. Organ. Behav. Manag. **23**(4), 1–20 (2005)
2. T Toledo, T Lotan, In-vehicle data recorder for evaluation of driving behavior and safety. Transport Res. Rec. J. Transport. Res. Board. **1953**(1), 112–119 (2006)
3. A Pérez, MI Garcia, M Nieto, JL Pedraza, S Rodríguez, J Zamorano, Argos: an advanced in-vehicle data recorder on a massively sensorized vehicle for car driver behavior experimentation. IEEE Trans. Intell. Transport. Syst. **11**(2), 463–473 (2010)
4. Lytx, Delivering insights. Driving results. www.lytx.com/. Accessed 04 Aug 2014
5. STS Thong, CT Han, TA Rahman, Intelligent fleet management system with concurrent GPS & GSM real-time positioning technology, in *Telecommunications, 2007. ITST'07. 7th International Conference on ITS* (IEEE, Sophia Antipolis, 2007), pp. 1–6
6. D Bekiaris, A Amditis, Advanced driver monitoring: the awake project, in *E-safety Congress and Exhibition* (Transportation Research Board (TRB), Lyon, 2002)
7. S Amin, S Andrews, S Apte, J Arnold, J Ban, M Benko, RM Bayen, B Chiou, C Claudel, C Claudel, T Dodson, O Elhamshary, C Flens-batina, M Gruteser, J-C Herrera, R Herring, B Hoh, Q Jacobson, T Iwuchukwu, J Lew, X Litrico, L Luddington, Margulici Jd, A Mortazavi, X Pan, T Rabbani, T Racine, E Sherlock-thomas, D Sutter, A Tinka, ed. by Citeseer, Mobile century using GPS mobile phones as traffic sensors: a field experiment, in *Proceedings of the 15th World Congress Intelligent Transport Systems: November 2008; New York* (World Congress Intelligent Transport Systems, New York, 2008)
8. C-Y Chan, On the detection of vehicular crashes-system characteristics and architecture. IEEE Trans. Veh. Tech. **51**(1), 180–193 (2002)
9. P Needham, Collision prevention: the role of an accident data recorder (adr), in *Advanced Driver Assistance Systems, 2001. ADAS. International Conference on (IEE Conf. Publ. No. 483)* (IET, Birmingham, 2001), pp. 48–51
10. MsFazeen, B Gozick, R Dantu, M Bhukhiya, MC Gonzalez, Safe driving using mobile phones. IEEE Trans. Intell. Transport. Syst. **13**(3), 1462–1468 (2012)
11. DA Johnson, MM Trivedi, Driving style recognition using a smartphone as a sensor platform, in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference On* (IEEE, Washington DC, 2011), pp. 1609–1615
12. P Mohan, VN Padmanabhan, R Ramjee, Nericell: Rich monitoring of road and traffic conditions using mobile smartphones, in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (ACM, Raleigh, 2008), pp. 323–336
13. C Saiprasert, T Pholprasit, W Pattara-atikom, Detecting driving events using smartphone, in *20th ITS World Congress* (World Congress Intelligent Transport Systems, Tokyo, 2013)
14. J Lin, E Keogh, S Lonardi, B Chiu, A symbolic representation of time series, with implications for streaming algorithms, in *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery* (ACM, New York, 2003), pp. 2–11
15. Symbolic Aggregate Approximation. http://www.cs.gmu.edu/~jessica/sax.htm. Accessed 04 Aug 2014

doi:10.1186/1687-1499-2014-135
Cite this article as: Chaovalit *et al.*: A method for driving event detection using SAX with resource usage exploration on smartphone platform. *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:135.