2012 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
MODELING & SIMULATION, TESTING AND VALIDATION (MSTV) MINI-SYMPOSIUM
AUGUST 14-16, MICHIGAN

DEVELOPMENT OF DRIVE CYCLES FOR FUEL CONSUMPTION EVALUATION OF MILITARY GROUND VEHICLES

Oguz H. Dagci
Andrew Cook
Phillip Shaw
AVL Powertrain Engineering Inc.
Plymouth, MI

ABSTRACT

This paper discusses the development of a methodology to generate drive cycles having a finite duration, but which are statistically representative of a larger set of usage data collected from fleet vehicles operating in the field. Given field-generated time vs. velocity data, acceleration at each data point is calculated, and each velocity and acceleration pair is binned using some calibrated level of fidelity. As a result, a velocity-acceleration matrix representing each vehicle operating point, as well as cumulative probability distribution functions for acceleration change and take-off acceleration are generated. These cumulative distribution functions are utilized to pick random velocity-acceleration pairs from the corresponding matrix, and the concatenation of each consecutive chosen velocity-acceleration pair constitutes the final drive cycle. Three drive cycles representing the high-, medium- and low-speed operation of the vehicle are generated from the field data, and these show measurable similar statistical characteristics to the initial master data superset. To validate the drive cycles, simulations were performed on a representative vehicle model using both the statistically generated drive cycles and the entirety of the field-collected dataset, and fuel economy results were found to be consistent between the two. Finally, a method for evaluating various use cases of the vehicle in terms of fuel economy using various combinations of these three drive cycles is discussed.

I. INTRODUCTION

The cost of fuel is a major expense for the US armed forces, especially when considering the need to transport it to various distant and often hostile locations around the globe. Since this elaborate fuel distribution infrastructure does not show any signs of shrinking or becoming less cost or labor intensive, improving fuel economy across the military fleet is seen as a necessity in order to achieve a reduction in cost and resource allocation [1].

Like any contemporary vehicle engineering effort, a cost-effective and results-oriented design approach for a military application requires the intensive simulation of several hardware permutations and operating scenarios. The two most important development resources for this early stage are an accurate vehicle model, and a drive cycle or series of cycles which adequately capture the real-world vehicle operating environment. Without first knowing how the vehicle is operated, it is difficult or impossible to accurately quantify any changes or improvements.

Since most ground vehicles will tend to be operated over a large range of terrains by drivers of varying ability and style, any vehicle simulation as part of a development process should take as many likely operating conditions into account as possible. Since simulating thousands of hours of collected data from several sources would be extremely computationally intensive, short-duration drive cycles should be created which will yield similar results to the full dataset, but run in a fraction of the time. The great challenge then, is to develop a drive cycle or suite of cycles which accurately represent a much larger dataset. Several methods which achieve this goal have been published, each with its own benefits and drawbacks given a particular application.

There are two major concepts for drive cycle generation. The first hinges on paring down and manipulating existing data into a microcosm of the larger dataset. In this case, the final drive cycle is simply built from short operating windows, called microtrips, taken from the larger dataset. The second method involves collecting a large amount of statistical information from the complete driving dataset, and generating a novel drive cycle which is completely independent of the original data, but characteristically similar, and therefore valid for use in a simulation exercise.

Several city-specific drive cycles have been developed using the microtrip method, specifically Sydney, Australia [2], Hong Kong, PRC [3], Pune, India [4], and Los Angeles, California [5]. Microtrip drive cycle generation has the benefit of being relatively straightforward computationally, and easy to validate. From the complete driving dataset, statistical information such as average speed, average acceleration and deceleration, maximum speed, idle time, etc. should all be collected along with road type [6], speed limit and terrain. Once this master data superset is cut up into microtrips consisting of individually distinct instances of driving such as highway, surface roads, etc., similar statistics can be collected for each microtrip. Each trip can then be compared to the initial data superset based on its measurable statistical content, and cycles can be built by combining multiple microtrips to build short-duration cycles with similar characteristics to the master dataset.

The main drawback to the microtrip-based drive cycle development method is that if the microtrips have too long of a duration, combining several of them may produce a synthetic drive cycle which is too lengthy to be useful for a quick simulation.

The appeal of the statistical drive cycle creation method is that the final cycles are completely decoupled from the master dataset, meaning that there is little to no restriction on duration. Furthermore, an infinite number of unique cycles can be developed from one single dataset, and the cycles are equally easy to validate against the master dataset as with microtrips.

Several studies have been conducted on the topic of statistical drive cycle development [7, 8, 9]. This method is very helpful when dealing with a dataset consisting of exceptionally long microtrips when compared to the desired length of the synthesized drive cycle. It is also beneficial when trying to remove identifiable driver behavior by combining all data produced by all drivers into a single data pool and drawing from it pseudo-randomly. The statistical drive cycle development method is chosen for this study due mainly to the length of the potential microtrips compared to the desired drive cycle duration.

This paper is organized as follows. First, the process of collecting vehicle operational data, and the pitfalls associated with instrumentation and processing are discussed in Section II. Cycle development methodology is outlined in detail in Section III, validation through simulation is discussed in Section IV, and the calculation of fuel economy when manipulating three drive cycles to mimic various vehicle usage cases is shown in Section V. The paper is concluded in Section VI.

II. DATA COLLECTION & PROCESSING

When developing a drive cycle or series of drive cycles to be used as microcosms of a much larger dataset, the usefulness of the cycles is found to be proportional to their statistical similarity to the larger set from which they are derived (see Validation and Simulation Results). For this reason, great care must be taken when collecting and processing the raw real-world data, as creating an accurate representation of incorrect or incomplete information will not yield useful results. Furthermore, since vehicle operating data is not typically collected by the same analyst who will eventually process it, especially when considering a fleet of several vehicles, specifically defining what information needs to be collected, and at what fidelity, will save many hours of unnecessary post-processing.

For the drive cycle development case outlined in this paper, a set of military ground vehicles were equipped with data acquisition hardware which collects vehicle speed both through the wheels and by GPS, along with various other vehicle details including engine speed, steering angle, etc. Although there are deviations, speed data is typically collected at a fidelity of one data point per second, and data acquisition typically begins with ignition key-on. With key-off, the collected data is saved into a file with a unique name representing the vehicle identification number, date, and various internal coding. In addition, since a vehicle could be operated and keyed on/off several times in a given day, each dated folder contains several independent vehicle runs, hereafter referred to as “trips”.

Since several vehicles may be instrumented by several engineers and technicians in various geographical locations, this exercise presents an excellent test case for exposing issues with inconsistent or incomplete data collection. While each vehicle may tend to produce very consistent data for each of its trips, the variation between vehicles could require a fair amount of post-processing to standardize everything into one large dataset.

Some potential inconsistencies which must be standardized to ensure efficient data processing in any collected set of data include units (miles vs. kilometers), naming conventions for the collected variables, and data collection fidelity such as data sampling time intervals.

Another consideration is incomplete data collection due to human error, sensor malfunction, or data logging failure. When large swaths of data must be processed, it is tempting to completely automate the process using a script or macro to combine multiple datasets and find averages or acceleration frequency content etc. However, a data collection error could generate a time vs. vehicle speed plot which has some reasonable duration and reasonable vehicle operating content, but is completely wrong and unrepresentative of what actually occurred. For instance, a sensor could be turned on and set to calibrate itself by outputting a constant 30mph signal for 5 minutes before operating normally, or a wire could be loose and every so often short to ground or an arbitrary voltage source to randomly output zero or some arbitrary non-zero speed. Looking at just the statistical content of a trip, the average speed may seem reasonable, and once the sensor settled, it would start correctly collecting vehicle acceleration and deceleration events. Not until an analyst actually takes the time to look at the data would it become apparent that for the first 5 minutes (or at arbitrary points) the data collected was plagued with sensor errors which distorted how the vehicle was actually being operated. This type of error could drastically skew the drive cycle statistics.

Finally, noise may exist in the data taking the form of either extremely short data collecting sessions, or very low speed sessions which clearly do not represent typical vehicle operation. For instance, a two hour driving trip with the vehicle mostly stationary, but occasionally moving at speeds less than 5mph can often be dismissed from the larger dataset as either periods of vehicle idle, or poor sensor calibration or data collection.

To quickly analyze all of the collected trips and eliminate any sensor errors which may produce a statistically reasonable trip consisting of unrepresentative data, a script was written to print the time-speed curve for each trip into its own image file. Each image is associated with the dataset naming convention mentioned above identifying vehicle/date/trip. With all of these image files populating a series of subfolders, an analyst can quickly click through each one, taking note of which appears to contain data errors (constant unwavering speeds, instantaneous speed changes, unreasonably short duration, excessive sensor noise etc.). Problematic trips are then scrutinized and thrown out or modified to eliminate portions containing obvious data errors. The definition of ‘data errors’ tends to be somewhat arbitrary and will vary for each application based on the amount of data available, the fidelity of this data, and what is considered reasonable operation within vehicle limitations.

With a dataset containing only error-free, high confidence data, the next step is to standardize the data collection interval. In the case of merging or comparing a trip with a 0.1s fidelity vs. a trip with a 1.0s fidelity, the three options are to either throw out 9/10 of the points collected in the 0.1s case, repeat each point of the 1.0s case 10 times to produce 10 identical points from 0.1 to 1.0, 1.1 to 2.0, 2.1 to 3.0 etc., or to interpolate such that the 9 points between 1.0s and 2.0s are a ramp instead of just duplicates. If data storage space is not an issue, it seldom makes sense to reduce the fidelity of a dataset. Every sensor will have noise, so a larger number of data points allows for this noise to be more easily filtered out. For instance, for a noise spike occurring at $t=2s$, if data were available for $t=1.9s$ and $t=2.1s$, it would be a lot easier to identify the spike as noise than if data only existed for $t=1s$ and $t=3s$. If the spike looks like the maximum vehicle acceleration between say $t=1s$ and $t=2s$, the same spike observed with a higher fidelity would look like 10x the maximum vehicle acceleration between $t=1.9s$ and $t=2s$. The higher fidelity data is therefore left untouched, and to avoid making any false assumptions about vehicle operation, the lower fidelity trips simply have their velocity points repeated several times to match the higher fidelity datasets (usually 10x as described in the above example). The reason for matching the fidelity is to simplify the data processing scripts when creating the actual drive cycles.

In order to use the drive cycle generation method discussed in subsequent sections, details of each trip need to be collected. These include average and maximum speed, average and maximum acceleration and deceleration, trip duration, time spent at idle (engine on, vehicle stationary), standard deviation of speed, and whether or not certain data like engine speed was being collected. Each trip is run through a script which mines all of these data, and places them into a sortable structure. Using the methodology described in this paper, manipulation of this structure is then used to determine what the synthetic drive cycles should look like with respect to speed, acceleration and idling content.

After conducting a detailed analysis of the entire dataset using just the statistics listed previously, three major classifications of trips become apparent: low speed, medium speed and high speed. Since the dataset for each drive cycle analysis exercise will be different given the vehicle type, fleet size, and operating conditions, the actual break points for classifying the low/medium/high subsets is moot in the context of this paper. However, it should be noted that the metrics used for classification include average speed, maximum speed, time spent idling, frequency of stop-and-go events etc. Furthermore, a single drive cycle could have been developed to represent the entirety of the raw dataset, but a single drive cycle would eliminate the degree of freedom needed to generate missions which represent various use cases of the vehicle.

III. METHODOLOGY

The methodology used in this paper for drive cycle synthesis is based on a concept presented in [7] and [8]. Building on these works, the limitations of the methodology and how they can be practically overcome are explained in detail here, and a new idle time assignment method is proposed. Furthermore, the synthetic drive cycles are validated by comparing not only statistical parameters, but also vehicle-level fuel economy simulations using the AVL-CRUISE Vehicle and Driveline Analysis Tool.

The proposed method requires the generation of three structures from the vehicle speed data:

- Launch Acceleration Array
- Acceleration Change Array
- Acceleration-Speed Matrix

Each index of the launch acceleration array corresponds to an acceleration value which is defined as the vehicle acceleration over one time step from standstill to any speed exceeding 0.1mph. The smallest non-zero vehicle speed of 0.1mph is a semi-arbitrary value which may be calibrated to prevent sensor noise from falsely indicating that the vehicle has been launched. In this case, all speed values below 0.1mph are considered vehicle standstill.

The launch acceleration array shows how many times the vehicle has launched at a given acceleration over the entire dataset. In this study, the resolution of launch acceleration and its maximum value are taken as $0.01 m/s^2$ and $2.50 m/s^2$ respectively, resulting in an array size of 250. Simply, each launch acceleration is binned using a fidelity of $0.01 m/s^2$, and then represented in the array as a single tally.

Similarly, the acceleration change array is constructed by assigning the number of occurrences of each acceleration change between consecutive time points in the speed data to its corresponding position in the acceleration change array. This array also uses a resolution of 0.01m/s^2. Since acceleration change can be negative, the size of the acceleration change array becomes 501 with the first and last elements in the array corresponding to -2.5m/s^2 and 2.5m/s^2 respectively, and the inclusion of 0m/s^2. Figure 1 shows these arrays graphically for clarity. In the next step, cumulative probability distribution functions (cdf) of the launch acceleration and acceleration change are derived from these arrays as seen in Figures 2 and 3 respectively.

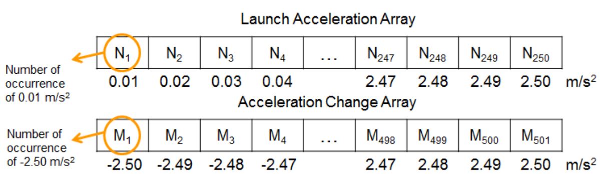

Figure 1: Acceleration Arrays

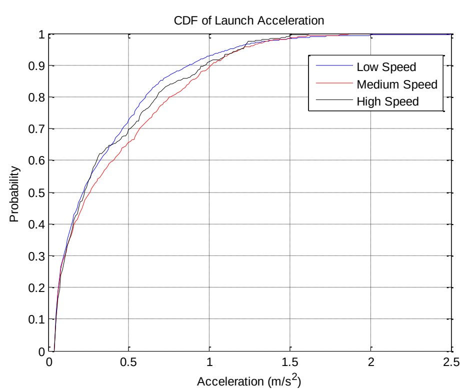

Figure 2: CDF of Launch Acceleration

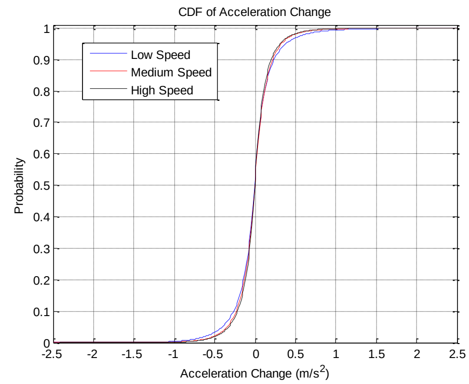

Figure 3: CDF of Acceleration Change

The final data structure required to synthesize drive cycles is the acceleration-speed matrix shown in Figure 4. The value in each cell of the matrix represents how many times a point with the corresponding acceleration and speed values occurs in the entire dataset. For this matrix, the resolutions of acceleration and vehicle speed were chosen as 0.01m/s^2 and 0.1mph respectively, and the fastest vehicle speed was set to 60mph, as this is the maximum speed achievable by the vehicle. The resolution of acceleration and vehicle speed in the acceleration arrays and acceleration-speed matrix were chosen such that the size of the arrays and matrix was reasonable for computing time and memory usage, and the data could also be processed with sufficient accuracy.

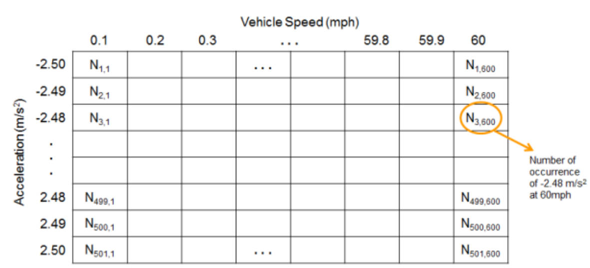

Figure 4: Acceleration - Speed Matrix

An acceleration-speed matrix used in one of the drive cycle design cases constructed as part of this work is illustrated in the form of a heat map in Figure 5. The horizontal center line represents 0m/s^2 acceleration, the far left represents 0mph, and the far right represents 60mph. As one would expect from a large vehicle, most acceleration events are relatively mild, and the majority of vehicle operation occurs below 45mph.

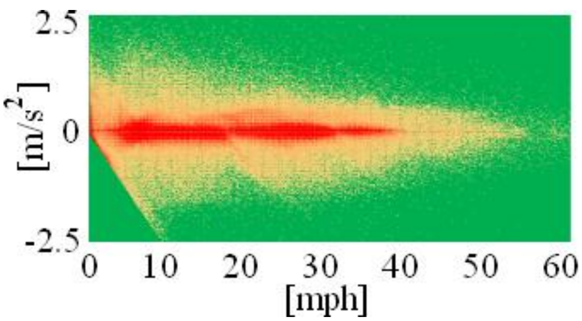

Figure 5: Accel-Speed Matrix, Heat Map

Algorithm Description

In this subsection, the drive cycle generating algorithm that processes the acceleration arrays and acceleration-speed matrix will be described step-by-step.

Before starting the algorithm execution, the desired duration of the drive cycle without idle time must be determined. This duration should be selected by engineering judgment such that it is long enough to represent the large vehicle speed database created by the collected data, but short enough to run both the cycle synthesis algorithm and eventually the vehicle simulations for which the cycle is being defined without overburdening computer or project timing resources. In the design case discussed in this paper, the synthesized drive cycle duration without idle time is chosen to be 40 minutes.

The algorithm starts while the vehicle is at standstill and is executed in the following order:

1- A uniformly distributed random number “Y” between 0 and 1 is generated. The launch acceleration “X” is calculated, given the cumulative probability of “Y”, from the launch acceleration cdf curve as seen in Figure 6. If the number of occurrences of this acceleration X is nonzero in the launch acceleration array, and X is large enough to let the vehicle speed exceed the 0.1mph threshold, X is accepted as a valid acceleration. The number of occurrence of this X in the array is then decreased by 1, and the new vehicle speed V_k is calculated as $V_k = X \cdot T$, where T is the time interval between data points. If the number of occurrences is zero, a new random number is generated, and the process is repeated until the occurrence of the corresponding launch acceleration is nonzero. As Figure 6 depicts, the purpose of generating a uniform random number and using the cdf curve in choosing a value for the launch acceleration, is to have higher probability of picking launch acceleration values that have occurred more often in the vehicle speed database.

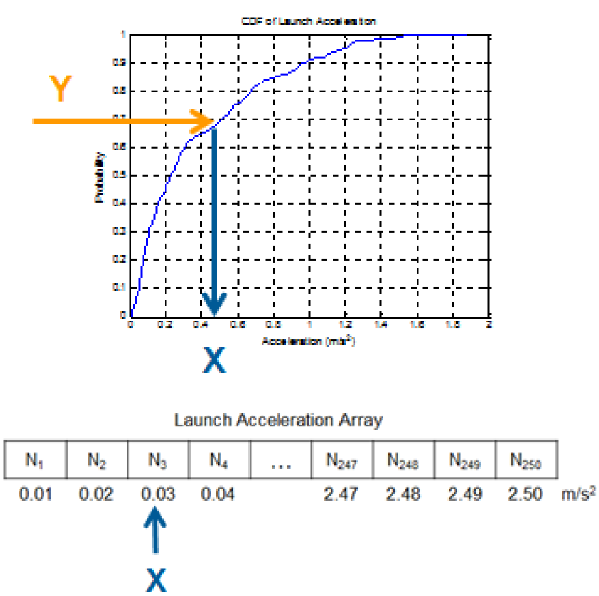

Figure 6: Selection of Launch Accel.

2- Another uniformly distributed random number "Z" between 0 and 1 is generated. The acceleration change "W" that gives the cumulative probability of "Z" is calculated from the acceleration change cdf curve as seen in Figure 7. If the number of occurrences of this acceleration change W is nonzero in the acceleration change array, and the new acceleration value stays inside the bounds of $2.5 m/s^2$ and $-2.5 m/s^2$, W is accepted as a valid acceleration change and the new acceleration a_k is calculated as $a_{k-1}+W$. If the number of occurrences of W in the acceleration change array is zero, a new random number is generated until the occurrence of the corresponding acceleration change is nonzero.

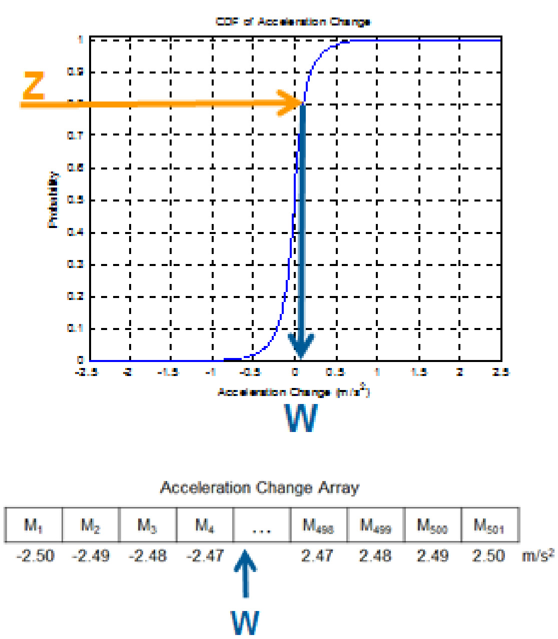

Figure 7: Selection of Accel. Change

3- Before the new (a_k,V_k) pair is included in the drive cycle, it should be validated with the Acceleration-Speed matrix shown in Figure 8 using the following logic:

Generating a drive cycle is similar to the concept of data compression, where a large dataset is represented by a much smaller set of data from which the larger dataset can be built. Therefore, in the drive cycle generation context, the term "compression ratio" can be defined as the ratio of the length of the entire dataset without idle time, over the length of a synthetic drive cycle without idle time. If the number of occurrences of (a_k,V_k) in its corresponding cell of the matrix is larger than or equal to the compression ratio, the (a_k,V_k) pair is a valid one and should be included in the drive cycle since this pair has been repeated frequently enough in the entire dataset to be included in the drive cycle. Once this pair is selected, the number of its occurrences in the matrix cell should be reduced by the compression ratio.

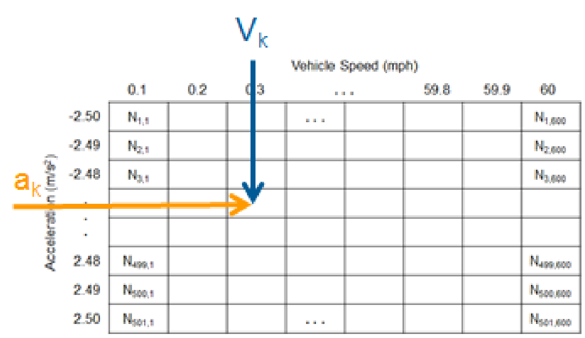

Figure 8: Selection of (a_k, V_k) Pair

If the number of occurrences of (a_k,V_k) pair is 0 in the matrix, the pair is not valid and step 2 in the algorithm description should be re-executed to generate a new (a_k,V_k) pair. This process is repeated until a valid (a_k,V_k) pair is found. If a valid pair cannot be found after a number of trials (500 was selected in this study), the algorithm gives a warning message and picks a new acceleration with nonzero occurrence in the column of V_k which is closest to the latest (a_k,V_k) pair. If the column of V_k does not hold any cell with a positive value (nonzero occurrence), the algorithm gives an error message and quits.

The algorithm description above shows the ideal case where the length of the collected dataset is much longer than the length of the drive cycle, and the data is spread uniformly enough in the acceleration-speed matrix that the number of occurrences of a large number of acceleration-speed pairs is higher than the compression ratio. However, in non-ideal cases, the dataset may not be long enough and the majority of cells in the matrix may contain values which are less than the compression ratio. In these circumstances, acceleration-speed pairs which occur more than some calibrated threshold value in the algorithm are considered valid pairs and are eligible to be used in the drive cycle. This threshold value is called the "occurrence threshold," and is set by running the algorithm using different threshold values. The value that tends to generate drive cycles with average vehicle speeds closest to the average speed of the entire dataset is chosen as the final threshold value. When an acceleration-speed pair with a number of occurrences greater than the threshold value and less than the compression ratio is selected, its occurrence in its corresponding cell is set to 0 so that it cannot be reelected in the remaining execution of the drive cycle generation algorithm. The algorithm would otherwise have generated a negative number, which is not useful in this drive cycle generation process.

Once an (a_k,V_k) pair is validated, k is incremented by one and the new V_k is calculated as $V_k = V_{k-1} + a_{k-1}\cdot T$.

4- If V_k is more than the calibrated zero-speed threshold, the algorithm goes back to step 2 to generate a new acceleration change and calculate the new a_k. If V_k is less than the zero-speed threshold, the algorithm goes to step 1 to select a launch acceleration value.

The algorithm execution described above continues until the length of the drive cycle reaches to the predetermined duration.

The generated drive cycle should be compared to the collected dataset to evaluate the degree of its representativeness with respect to some statistical terms [7],[2],[9]. The terms used in this comparison step are as follows:

- Average Speed without idle
- Root-mean-square (rms) of acceleration
- Average Positive Acceleration
- Average Negative Acceleration
- Coefficient of Variation (CoV)
- Positive Kinetic Energy (PKE)
- % Time between 0.1mph – 10mph
- % Time between 10.1mph – 20mph
- % Time between 20.1mph – 30mph
- % Time between 30.1mph – 40mph
- % Time between 40.1mph – 50mph
- % Time between 50.1mph – 60mph

If the generated drive cycle is far off from the reference dataset, it is discarded and the algorithm is run again until a drive cycle statistically representing the raw data is generated.

Idle Time Assignment

The algorithm description above generates a drive cycle without idle time. The idle periods should be inserted to each drive cycle according to the following logic:

First, the total idle time duration is calculated from the collected raw dataset. The percentage of the total idle time in the drive cycle should match that of the raw data. The challenge however, is how to appropriately distribute the total idle time amongst the vehicle zero-speed points throughout the generated drive cycle.

The proposed solution to this problem is to generate a cdf of the idle periods in the collected dataset. The raw dataset includes many trips, each containing multiple idle periods. The duration percentage of each idle period in each trip is calculated. From this data, a histogram and cdf of the idle period percentages are created as seen in Figures 9 and 10 respectively.

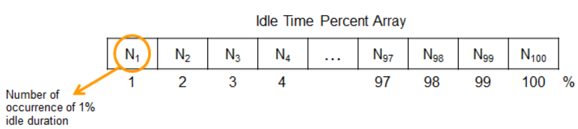

Figure 9: Idle Period Array

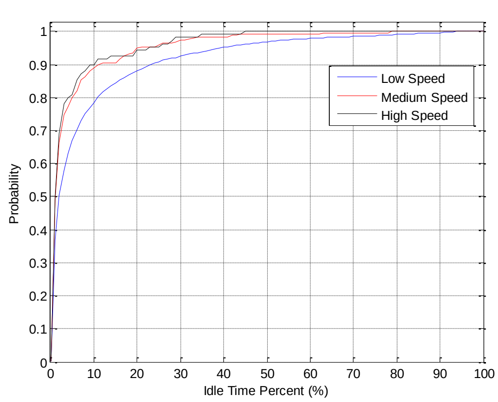

Figure 10: CDF of Idle Period Percentage

In the next step of the algorithm, the number of zero-speed crossings "M" in the generated drive cycle is counted. A zero-speed crossing event occurs when the vehicle speed goes below the zero-speed threshold described in Step 4 of the drive cycle generation algorithm. Since the duration of the drive cycle without idle time "N" is fixed and the percentage of total idle time in the raw data "P" is known, the total idle time "I" in the drive cycle can be easily calculated using equation (1). As a result, the idle time assignment algorithm should determine the duration of "M" idle segments whose sum will be equal to "I".

$$I = \frac{N}{(1-P)} \cdot P \qquad (1)$$

The idle time assignment procedure is similar to the methodology for generating launch acceleration and acceleration change values. First, a uniformly distributed random number "Y" between 0 and 1 is generated. The idle time percent "X" corresponding to "Y" is then found from the cdf curve of the idle time percent. If the number of occurrences of "X" is nonzero in the idle time percentage array (see Figure 11), "X" is a valid idle time percentage and the duration of the idle segment is calculated using equation (2). This procedure is repeated until the total number of idle segments is equal to "M" and the total duration of all "M" idle segments exceeds the desired idle time of "I". If the sum of "M" idle segment durations is less than "I" or the sum of idle segment durations exceeds "I" before the number of idle segments reaches "M", the algorithm resets and starts over again. This search continues until the algorithm finds "M" idle segments whose total duration is greater than "I" and the sum of the durations of the first "M-1" segments is less than "I". Once the search is successful, the duration of the last (M^{th}) idle segment is reduced such that the total idle duration is equal to "I".

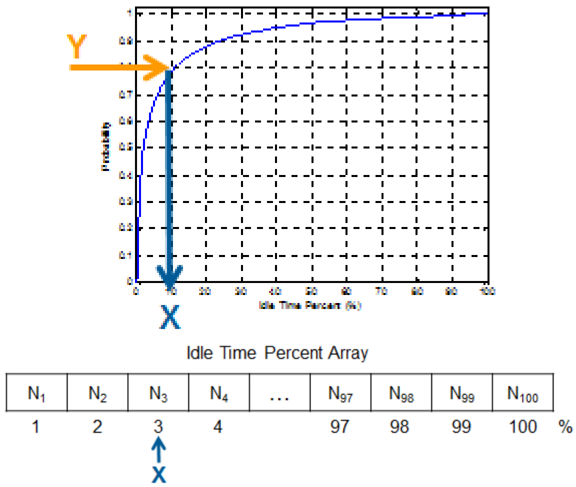

Figure 11: Selection of Idle Time

$$\textit{Duration of Idle Period} = \frac{0.01 \cdot N}{(1-P)} \cdot X \qquad (2)$$

After the idle time assignment is complete, the fuel economy numbers for the drive cycle and raw data are calculated using the target vehicle model. The drive cycles which best represent the raw data in terms of fuel economy and other statistical parameters listed in the previous subsection are accepted as valid drive cycles.

IV. IMPLEMENTATION AND VALIDATION OF RESULTS

Generating one drive cycle for each of the three trip types enables the creation of composite drive cycles, each a harmonically weighted average of the fuel economy numbers of the three generated drive cycles [10].

In this paper, only the drive cycle generated from the high speed trip group is shown for brevity, since the low and medium speed drive cycles are generated in a similar manner.

The duration of the concatenated trips belonging to the high speed group was 3800 minutes, of which 32% is idle time. As a first step, the duration of the drive cycle was determined as 40 minutes without idle time. This number should be chosen with engineering judgment considering the total length of the dataset, trip durations within the dataset, and the duration of several typical drive cycles in the automotive industry. According to the definition in the previous section, the compression ratio was calculated as $65 \left(\frac{3800\,\text{min}}{40\,\text{min}}(1-0.32) = 65 \right)$, which means that an acceleration-speed pair in the acceleration-speed matrix which occurred 65 times in the high speed dataset needs to occur just once in the corresponding drive cycle. Several runs of the drive cycle generation algorithm indicated that the zero-speed threshold and the occurrence threshold which determines the validity of an acceleration-speed pair should be 1mph and 6 respectively, in order to accurately represent the high speed dataset.

The most important methodology validation step is to confirm that the simulated vehicle achieves identical fuel economy figures when running over the entirety of the raw dataset, as when running over the shortened synthetic drive cycle. As the intention of creating a synthetic drive cycle is to reduce processor time when analyzing vehicle model changes, it should be expected that the validation step is very processor intensive, especially when considering an extremely large field-collected dataset. The only way to effectively validate the synthetic drive cycle is therefore to run the complete dataset and obtain a baseline for comparison. From a drive cycle validation perspective, even if the synthetic drive cycle does not produce the expected fuel economy numbers, the full dataset will only need to be simulated once and can be continually used as a benchmark against all future work, assuming the vehicle model does not change.

Using the AVL-CRUISE Vehicle System and Driveline Analysis Tool, a baseline vehicle was constructed. Considering that a synthetic drive cycle should be statistically representative of a larger dataset, the actual vehicle model is of little importance. Any vehicle model should be able to run through the entire raw dataset, and then the synthetic cycle, and yield similar fuel economy numbers given steady state operating conditions, and the scaling of any technologies such as start-stop or energy storage which may show a benefit over lengthier or shorter simulations.

The model was first simulated over the full high speed dataset and the average fuel rate and fuel economy numbers were recorded. Next, the same model was simulated using the several synthetic high speed drive cycles with no idle time integrated into the cycle, but with 32% idle time added to the end, representing the 32% key-on time that the vehicle spends idling over the complete high speed dataset. For completeness, all of these simulations were carried out with the vehicle empty, at half rated weight capacity, and at full rated weight capacity. The drive cycle with the average fuel rate and fuel economy closest to those of the full high speed dataset at all three weight settings was chosen as the final, most accurately representative drive cycle, shown in Figure 12.

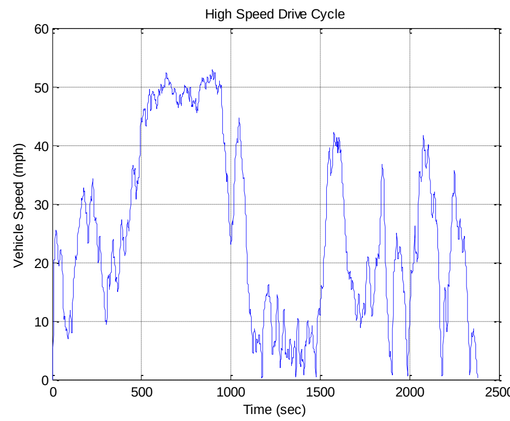

Figure 12: HS Drive Cycle without Idle

For the final stage of the drive cycle selection, idle time segments were assigned to the chosen drive cycle in Figure 12. As seen in the drive cycle, there are 9 points where vehicle speed crosses the zero-speed threshold. In order to match the high speed dataset, the total idle time should be 32% of the complete drive cycle. Since the duration of the drive cycle without idle time was selected to be 40min (2400sec), the idle time in the drive cycle can be calculated as 1129sec using equation (2). The idle time assignment algorithm described in the previous section searches for 9 idle segments whose sum is 1129sec and which comply with the cdf of idle time percent. These idle segments are then assigned to the locations of the 9 zero-speed points in the drive cycle, and Figure 13 depicts the final drive cycle with the idle segments.

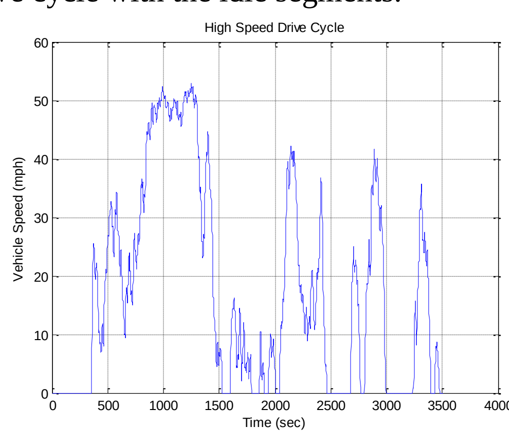

Figure 13: High Speed Drive Cycle

In Table 1, this synthetic high speed cycle is compared to the complete high speed dataset in terms of several statistical parameters, average fuel rate, and fuel economy for a half loaded vehicle.

As seen in Table 1, a 59-minute high speed drive cycle (2400s + 1129s) can represent the full 3800-minute high speed dataset with a very high level of confidence. The simulation outlined in Table 1 considers the vehicle at half rated weight capacity and results in a fuel economy error of 0.9% compared to the full dataset baseline. The simulations carried out for empty and full vehicle loading also yield very reasonable fuel economy results, with error percentages of 3.8% and -0.8% respectively.

The fuel economy and average speed for several synthetically generated drive cycles representing the Low, Medium and High speed datasets is shown in Table 2. Unlike the high speed case discussed above, these will not be analyzed in detail, but have been included for completeness. It should be apparent that while a close average speed will likely yield a close fuel economy, these two values by no means have a linear relationship. The same can be said for all of the statistics in Table 1, and it is therefore important to consider several characteristics of a given cycle before deciding to run any resource-intensive vehicle-level simulations to measure fuel economy. Further, it should be noted here that when analyzing the Low and Medium cycles, compression ratio values of 392 and 135 were derived respectively.

Table 1: Comparison of High Speed Drive Cycle to the High Speed Dataset

	High Speed Dataset	**High Speed Drive Cycle**
Ave. Speed	25.0 mph	25.2 mph
Idle Percent	32%	32%
CoV of Speed	0.55	0.61
Duration	3800 min.	59 min.
rms of Acceleration	0.28 m/s^2	0.27 m/s^2
Ave. + Acceleration	0.19 m/s^2	0.215 m/s^2
Ave. – Acceleration	0.20 m/s^2	0.21 m/s^2
PKE	0.17 m/s^2	0.18 m/s^2
% Time btw 0.1-10mph	19%	20%
% Time btw 10.1-20mph	21%	22%
% Time btw 20.1-30mph	19%	21%
% Time btw 30.1-40mph	26%	14%
% Time btw 40.1-50mph	14%	16%
% Time btw 50.1-60mph	1%	6%
Ave. Fuel Flow	2.42 gph	2.40 gph
Error		-0.8%
Fuel Econ.	7.04 mpg	7.10 mpg
Error		0.9%

Table 2: Similarity of Synthetic Low, Med. & High Speed Drive Cycles to Raw Datasets

Data Subset	**Average Speed**		**Mpg**	
Low Speed	9.4 mph		2.38	
Medium Speed	16.6 mph		5.00	
High Speed	25.0 mph		7.04	
Cycle	**Avg Spd**	**Error**	**Mpg**	**Error**
Low 01	9.191	2.2%	2.48	4.2%
Low 02	9.194	2.2%	2.50	5.1%
Low 03	9.168	2.5%	2.33	2.1%
Cycle	**Avg Spd**	**Error**	**Mpg**	**Error**
Med 01	16.516	0.5%	5.18	3.7%
Med 02	16.439	1.0%	5.16	3.2%
Med 03	16.111	2.9%	5.05	1.1%
Cycle	**Avg Spd**	**Error**	**Mpg**	**Error**
High 01	24.924	0.3%	7.02	0.2%
High 02	24.602	1.6%	6.89	2.1%
High 03	25.233	0.9%	7.10	0.9%

V. MISSION DEVELOPMENT

The usefulness of generating three different drive cycles from one master database is the ability to calculate the fuel economy of the vehicle for various use cases, called missions for this application. Equation (3) is used to obtain the composite fuel economy of a mission from the combined, weighted fuel economy results of the three drive cycles. Each weight in equation (3) represents what percentage of the total time the vehicle travels in the corresponding use case. For instance, if W_1 is chosen as 0.2, the vehicle travels 20% of its time in the high speed mode. The benefit of this method is to run just three simulations (one for each drive cycle) and then calculate the fuel economy of a potentially infinite number of missions by changing the weights during post-processing without further conducting any simulations.

$$Mission\,FE = \frac{1}{\dfrac{W_1}{FE_{hs}} + \dfrac{W_2}{FE_{ms}} + \dfrac{W_3}{FE_{ss}}} \qquad (3)$$

$$W_1 + W_2 + W_3 = 1$$

hs: High Speed
ms: Medium Speed
ss: Slow Speed

VI. CONCLUSION

In this paper, three types of drive cycles are developed, each representing trips with different average vehicle speed and idle time percentages. The algorithm that generates the drive cycles is described in sufficient detail with references to give the reader a comprehensive understanding of the methodology. The generated drive cycles are compared to the field data statistically. Furthermore, it is shown through vehicle simulations that the fuel economy for the reference dataset can be predicted by the synthesized drive cycles within a reasonable error percentage. As a result, these drive cycles enable the efficient and accurate evaluation of fuel economy improving technologies for several different use cases of a specific military ground vehicle.

The next steps in the drive cycle development are to assign and integrate terrain data and steering to the drive cycles. Vehicle simulations will also be conducted with various combinations of fuel economy-improving technologies to determine which mix of technologies will maximize the fuel economy/cost ratio for each defined mission.

REFERENCES

[1] United States Government Accountability Office, "DOD Needs to Increase Attention on Fuel Demand Management at Forward-Deployed Locations", GAO-09-300, February 2009.

[2] J. H. Kent, G. H. Allen, G. Rule, "A Driving Cycle for Sydney", Transportation Research, Volume 12, pp. 147-152, 1978.

[3] W. T. Hung, H. Y. Tong, C. P. Lee, K. Ha, L. Y. Pao, "Development of a Practical Driving Cycle Construction Methodology: A Case Study in Hong Kong", Transportation Research Part D: Transport and Environment, Volume 12, Issue 2, pp. 115-128, 2007.

[4] S. H. Kamble, T. V. Mathew, G. K. Sharma, "Development of Real-world Driving Cycle: Case study of Pune, India", Transportation Research Part D: Transport and Environment, Volume 14, Issue 2, pp. 132-140, 2009.

[5] T. C. Austin, F. J. DiGenova, T. R. Carlson, R. W. Joy, K. A. Gianolini, J. M. Lee, "Characterization of driving patterns and emissions from light-duty vehicles in California", The Final Report, Contract No: A932–185, California Air Resources Board, Sacramento, 1993.

[6] S. Fincher, C. Palacios, S. Kishan, D. Preusse, H. Perez, "Modifying Link-Level Emissions Modeling Procedures for Applications within the MOVES Framework", The Final Report, Contract No: DTFH61–09–C–00028, Federal Highway Administration, U.S. Department of Transportation, September 2010.

[7] T. J. Lyons, J. R. Kenworthy, P. I. Austin, P. W. G. Newman, "The Development of a Driving Cycle for Fuel Consumption and Emissions Evaluation", Transportation Research Part A, Volume 20A, No. 6, pp. 447-462, 1986.

[8] A. Alessandrini, F. Orecchini, "A Driving Cycle for Electrically-Driven Vehicles in Rome", Proceedings of the Institution of Mechanical Engineers, Volume 217, Part D: Journal of Automobile Engineering, pp. 781-789, 2003.

[9] Z. Dai, D. Niemeier, D. Eisinger, "Driving Cycles: a new cycle-building method that better represents real-world emissions", University of California, Task Order No. 66, September 18, 2008.

[10] "Fuel Economy Labeling of Motor Vehicle Revisions to improve Calculation of Fuel Economy Estimates", Final Technical Support Document, EPA420-R-06-017, U.S. Environmental Protection Agency, 2006.