FACILITATION OF VISUAL PATTERN RECOGNITION BY EXTRACTION OF RELEVANT FEATURES FROM MICROSCOPIC TRAFFIC DATA

A Thesis

by

MATTHEW JAMES FIELDS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2007

Major Subject: Computer Science

FACILITATION OF VISUAL PATTERN RECOGNITION BY EXTRACTION OF RELEVANT FEATURES FROM MICROSCOPIC TRAFFIC DATA

A Thesis

by

MATTHEW JAMES FIELDS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Paul Nelson
Committee Members,	Ricardo Gutierrez-Osuna
	Gene Hawkins
Head of Department,	Valerie E. Taylor

December 2007

Major Subject: Computer Science

ABSTRACT

Facilitation of Visual Pattern Recognition by Extraction of
Relevant Features from Microscopic Traffic Data. (December 2007)

Matthew James Fields, B.S., Trinity University
Chair of Advisory Committee: Dr. Paul Nelson

An experimental approach to traffic flow analysis is presented in which methodology from pattern recognition is applied to a specific dataset to examine its utility in determining traffic patterns. The selected dataset for this work, taken from a 1985 study by JHK and Associates (traffic research) for the Federal Highway Administration, covers an hour long time period over a quarter mile section and includes nine different identifying features for traffic at any given time. The initial step is to select the most pertinent of these features as a target for extraction and local storage during the experiment. The tools created for this approach, a two-level hierarchical group of operators, are used to extract features from the dataset to create a feature space; this is done to minimize the experimental set to a matrix of desirable attributes from the vehicles on the roadway. The application is to identify if this data can be readily parsed into four distinct traffic states; in this case, the state of a vehicle is defined by its velocity and acceleration at a selected timestamp. A three-dimensional plot is used, with color as the third dimension and seen from a top-down perspective, to initially identify vehicle states in a section of roadway over a selected section of time. This is followed by applying *k*-means clustering, in this case with *k*=4 to match the four distinct traffic states, to the feature space to examine its viability in determining the states of vehicles in a time section. The method's accuracy is viewed through silhouette plots. Finally, a group of experiments run through a decision-tree architecture is compared to the *k*-means clustering approach. Each decision-tree format uses sets of predefined values for velocity and acceleration to parse the data into the four states; modifications are made to acceleration and deceleration values to examine different results.

The three-dimensional plots provide a visual example of congested traffic for use in performing visual comparisons of the clustering results. The silhouette plot results of the *k*-means experiments show inaccuracy for certain clusters; on the other hand, the decision-tree work shows promise for future work.

ACKNOWLEDGMENTS

First, I thank Dr. Henry C. Lieu, of the Federal Highway Administration, for providing a copy of the JHK datasets that were integral to the work presented in this thesis.

Next, I would like to thank my graduate advisor, Dr. Paul Nelson, for his ongoing support in this path to my graduate degree. Also, to the committee members, thank you for your input and assistance with this research. The Computer Science Department has been instrumental in guiding this document through to publication.

Additional thanks and appreciation are owed to my parents, Rose Ann and Michael Fields.

And finally, thank you to my future wife, Lindsey K. Moore for your constant encouragement, love and support.

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

Page

LIST OF TABLES

1. INTRODUCTION

1.1 – Objective

Moving vehicles tend to fall, at any given time, into one of four states as a result of road conditions, number of motorists, weather, and other factors. The labels commonly given these four states are as follows. *Free flow* designates travel at roughly the speed desired by the driver, with negligible acceleration. *Congested flow* designates stunted travel speeds, with negligible acceleration; a “traffic jam” is an extreme case of congested flow. An *acceleration wave* denotes acceleration to return to free flow speeds , such as when exiting a traffic jam; in the field of transportation science and engineering traffic “backed up” at a bottleneck is often termed a “queue,” and in that context an “acceleration wave” may be termed as “queue discharge.” A *deceleration wave*, or *shock wave*, denotes a rapid deceleration of traffic, such as a group of motorists approaching a traffic jam. These four states of traffic flow are recognizable to experienced motorists. The first two are more-or-less embedded in the standard “level of service” measure of effectiveness for roadway facility performance, as described by the Highway Capacity Manual [Transportation Research Board 2000]. All four of these states are identifiable within the Lighthill-Whitham [1955]-Richards [1956] kinematic-wave “fluid-like” model of vehicular traffic flow, which was the earliest mathematically predictive model of vehicular traffic flow.

This thesis follows the style of *ACM Transactions on Mathematical Software.*

Notwithstanding these intuitively agreeable categories, there seems to be a paucity of accepted practical methods for the processing of traffic data to permit ready identification of spatiotemporal regions within which one of these traffic states dominates. *The objective of this thesis is to apply methods from the field of pattern recognition to facilitate the identification, from visual representations of the data, of spatiotemporal regions within which one of these four states of traffic flow prevails*. I introduce a conceptual framework, and a set of tools designed for extracting information from microscopic traffic-flow data that have been implemented as MatLab® code. The techniques applied from pattern recognition include:

- feature extraction and supervised learning, particularly *k*-means clustering with *k=4* and intial selection of centroids informally by the preceding "state" characteristics, and
- a decision tree methodology that is used in comparison to the *k*-means clustering.

The techniques discussed are specific to *microscopic* data, by which is intended data recording the motion of individual vehicles as a function of time over a designated section of roadway. The expectation is that the insights gathered from this exercise will lead to improved methods of analysis for the more widely available but lower-quality macroscopic loop-detector data. It should be noted that this thesis is an expanded work to the paper and presentation given at the Transportation Research Board's Annual Meeting in Washington, D.C., during the month of January 2006.

Following is an outline of this thesis. The following Section 1.2 of this introductory section is a summary of prior efforts at identifying spatotemporal regions by the class, in some sense, of the traffic flow prevailing therein, principally for the purpose of positioning the present effort within the matrix of prior efforts. In Section 2 I describe an abstract view of pattern recognition that is adapted from the literature to provide a detailed framework for the application of pattern recognition to the objective described above, specifically the assignment of a vehicle at a given time to one of the four kinematic-wave classes enumerated in the preceding section. Section 3 is devoted to:

- presentation of an abstract representation of microscopic traffic-flow data sets that fits into the pattern recognition framework of Section 2 (Section 3.1);
- descriptions of two well-known collections of microscopic traffic-flow data sets, and their mapping into the abstract representation of Section 3.1 (Section 3.1.1 and 3.1.2);
- discussion of a library of operators, to act upon abstract microscopic traffic-flow datasets, that I have designed, and implemented in MatLab®; and
- visual macroscopic examples of a traffic dataset, including the commonly used trajectory plot.

Section 4 contains a summary of the generalities of *k*-means clustering, including how it is employed in the code, for pattern recognition applied to kinematic-wave classification for microscopic traffic-flow data sets. I will also mention and give an example of silhouette plots, which play a part in analyzing results from applying *k*-means. In Section 5, I describe how *k*-means clustering is used in various examples to create centroids for experiments on a target roadway section. Section 6 presents how the resulting centroids are applied to the dataset, with regard to the specific section; furthermore, the resulting graphical and numerical descriptions of the resulting classes after parsing are given, with subsequent attempts to identify kinematic-wave spatiotemporal regions from plots of the results from this microscopic classification. Section 7 contains a similar preliminary application of decision trees for the pattern recognition, along with a comparative discussion of the results to those obtained using the *k*-means clustering. My concluding Section 8 contains a summary of the results, and recommendations for further related work.

1.2 – Traffic-flow Measurements and Related Spatiotemporal Classification Schema

This is a thesis in applied computer science, with the application domain being vehicular traffic flow. It is directed toward developing tools capable of leading to a higher level of understanding of data, especially microscopic data, for vehicular traffic flow, but it also is informed by a variety of theories of vehicular traffic flow that have been developed over the past 50-75 years. Most of these theories were developed in an effort to better understand the basis for classification schema that were themselves developed in an effort to understand a variety of traffic-flow data. The objective of this section is to describe these various classification schema, and the data and data analyses underlying them, so as to delineate how the present work fits into the existing body of knowledge, within this application domain. I defer to the references cited for explications of the associated mathematical theories. I also briefly mention some possibilities, not pursued in this thesis, of applying recent and ongoing developments within computer science to the application domain of vehicular traffic flow.

I begin with a brief description of the prevalent types of measurements of vehicular traffic flow, which is to say sources of data. By far the most common of these is loop-detector measurements, which are provided by single or closely spaced pairs of inductive loops embedded in roadways. These detect the presence or absence of vehicles directly overhead at some relatively high frequency of sampling, typically 60 cycles per second in North America. Single-loop detectors can determine the number of vehicles passing per unit time (flow) and the percentage of time a vehicle is overhead (occupancy), but can provide speeds only if a vehicle length is assumed; paired-loop detectors can provide speeds additionally. Loop detectors typically are spaced from ½ to one mile apart, so sample relatively coarsely in space. They sample rather frequently in time, but the data typically are stored only in some time-aggregated form (e.g., average occupancy over 20-second intervals, or mean speed over one-minute intervals), so that the net result is a rather coarse sampling in both time and space. However loop-detector data are widely available, because they are relatively inexpensive, and have been installed on freeways over much of the developed world to provide traffic management centers real-time information about the state-of-affairs on roadways.

By contrast microscopic traffic-flow data typically come from photographs taken from aircraft or stationary elevated objects. These are much higher quality, in the sense of providing samples at a much higher frequency in both time (depending on the frame rate) and space. Optical considerations (angle of view) limit photographic data to relatively short sections of roadway. Further such data are typically much more expensive to obtain than loop-detector data, especially in regard to digitization of the data. Although, there are newer forms of data gathering being researched, such as the efforts of the National Consortium on Remote Sensing in Transportation [2007].

Transportation science work is often informed by the kinematic-wave model (KWM) [Lighthill and Whitham 1955] [Richards 1956]. This was the original quantitative model for traffic flow, and it reproduces the four classes of traffic flow described in the preceding Objectives section. Notwithstanding this and the intuitive nature of those classes of traffic flow, tools for the automated classification of a given traffic flow pattern into one of these classes have not yet been developed. The proposed work is intended as a step in that direction.

Analyses of data from loop detectors have raised questions as to the validity of this KWM classification scheme [Drake et al. 1967] [Koshi et al. 1981], especially in their failure to confirm the Greenshields hypothesis [1934], which asserts the existence of a bivariate relationship between vehicular flow (vehicles per unit time) and density (vehicles per unit length). As the Greenshields hypothesis is one of the two major components of the Lighthill-Whitham [1955]-Richards [1956] model,[1] this casts doubt on that model, and leaves intuition as the only remaining basis of support for the four-state classification scheme. However, more recent reviews [Cassidy 1998] suggest the possibility that the classical failures to validate the Greenshields hypothesis could be due to flawed techniques for analyzing the data. Here we interpret those results as

[1] The other is conservation of vehicles.

motivation to attempt a direct validation of the four-state KWM classification scheme. The tools we are led to develop could also be applied toward direct validation of the Greenshields hypothesis, although that is outside the scope of the presently proposed work.

Alternatives to the KWM traffic-flow classification scheme exist, some of them motivated by the apparent failures of KWM, as described above. Certainly the most familiar of these to practicing transportation engineers is the traditional level of service (LOS) classification, according to which: "six LOS are defined for each type of facility that has analysis procedures available. Letters designate each level, from A to F, with LOS A representing the best operating conditions and LOS F the worst" [Transportation Research Board 2000].

Yet another classification scheme has been offered by B. S. Kerner and co-workers; the essence of this classification scheme seems to be well-captured by the following quotation: "There are three traffic phases: (1) free flow, (2) synchronized flow, and (3) wide moving jam" [Kerner and Klenov 2002]. Note however that a more recent work [Kerner 2004] seems to suggest a substantial variety of subclasses of these three apparently basic classes. Finally, Schönhof and Helbing [2004] have suggested a classification of traffic flow on a 30 km stretch of German freeway into five classes (pinned localized clusters, homogeneous congested traffic, oscillating congested traffic, stop-and-go waves and moving localized clusters), based on application of an adaptive smoothing method for multiple dual-loop detector data aggregated over one-minute intervals..

Regarding these various proposed classification schemes, note first that the validity or invalidity of one such classification scheme is *a priori* quite independent of the validity or invalidity of the other; that is, all may be valid, all may be invalid, or there may any combination of these. However the differing vocabularies, and lack of precise definitions of the various classes in terms of some common set of attributes ("features," in the terminology of the following section), makes it difficult to achieve any common understanding.

Second, I observe that the LOS classes seem to be sufficiently well-defined, in terms of attributes that can be extracted from measurements for a particular type of facility, so that one could, subject to availability of appropriate data, develop a corresponding pattern recognition algorithm (particularly a classifier), in the sense of the elements of pattern recognition as outlined in Section 2 below. While we have no doubt that the classifications schemes of Kerner and Klenov [2002], Kerner [2004], and of Schönhof and Helbing [2005], as cited above, have significant merit, it does not yet appear that they are yet sufficiently well-developed so as to permit application of the formal methods of pattern recognition. More specifically, in the terminology of Section 2 below, it is not clear how one could view the collections of classes suggested by these workers as comprising a partition of some feature space having attributes that are algorithmically defined in terms of certain measurements.

I close by noting that a great deal of work in computer science has been motivated by spatiotemporal identification within images ("object recognition"). However, very little work in computer science has been directed toward identification of spatiotemporal regions within which the moving vehicles predominantly share some characteristic. The lone exception to this that I find in the literature is the work of Shahar and Molina [1998]. That framework is not employed in the present work, nor does there seem to have been significant further development in the traffic-flow application domain that builds upon it; however it does appear to offer a promising basis for such developments.

2. ELEMENTS OF PATTERN RECOGNITION

Pattern recognition is a set of tools, using varied degrees of automation or human "supervision," in applications in research analysis and practice to identify, recognize, and verify classes that the information in question can exist in. It should be noted that "class," as in classification, is synonymous with anything that can be described as a pattern, state, phase, or similar concept. For clarity in this thesis, the word "class" will be used; furthermore, the four states of vehicular traffic flow listed in the Objective section are the primary instances of classes that are of interest here. Patterns can be parsed into many classes; it is the job of performing pattern recognition to quantify the degree to which a classification fits particular patterns. Pattern recognition is best described in the form of stages that perform necessary tasks to form an overall algorithm. Webb [2002] uses a multistage description that can be partially summarized as sensor, representation pattern, feature extraction/selector, feature pattern, classifier, and decision. These stages perform the tasks of gathering, examining, and analyzing data for purposes of interpretation of results.

The first step in application of pattern recognition to classification is designing a problem that requires classification. Some applications of pattern recognition include diagnosis of medical conditions or weather patterns [Webb 2002]. Techniques are used to create systems, train them using test data, and assess the usefulness of the creation versus the proposed goal of the system. For the weather system, using past data and known outcomes from weather patterns, the developers can determine if the machine learning system is predicting correctly or if modification and retraining is necessary.

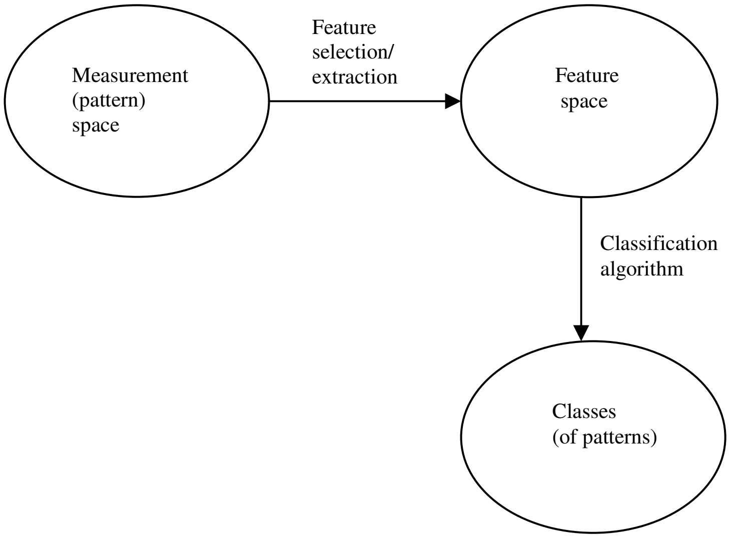

Diagram 1

Diagram 1 gives an example view of how the pattern recognition process is applied in this work. The five stages, listed in order of operation, are elaborated on and explained in the following paragraphs.

The process begins with data collection in order to create a measurement space for examination. The examination is performed to construct initial ideas for the measurement space as well as identify the need for potential data cleaning. Features within the measurement space are identified, pulled out, and separated for closer evaluation. The evaluation will determine the usefulness of the feature space, the robustness of the method used, or if steps in the process need to be performed once again.

Selecting and retrieving relevant data from measurements are tied to *feature selection and extraction*, respectively. It is important to develop a robust feature space, as Fu et al. [1970] explain in their paper. “One important problem in pattern recognition is the selection of effective features from a given set of feature measurements. It is known that the performance of a pattern recognition system is closely related to the feature measurements taken by the classifier.”

Young and Calvert [1974] provide an initial framework for feature space development. *Features* are an *M*-Dimensional set of data, Ω_y, that is taken from an *N*-Dimentional pattern space, Ωx. In their book, they take special care to emphasize that *features* and *measurements* are not one in the same. “The *selection of measurements* is based on our prior knowledge or experience on the particular pattern recognition problem . . . Feature extraction or selection is, on the other hand, essentially a scheme that reduces the dimensionality from *N* to *M*” [Young and Calvert 1974]. They also note that the reduction in dimensionality leads to an obvious loss of certain information from the pattern space. Two justifications for feature extraction given by the authors are relevant to the proposed work; specifically, that the feature space is more meaningful than the measurement space and that prior knowledge of the redundancy and correlation of measurements allow us to reduce the dimensionality of the measurement space without losing much information.

The first point is especially relevant since traffic measurements often include numerous fields of data that would be useful only for certain analyses. While having this unfiltered traffic data directly in the dataset is nice, a system that can compute new desirable features from data stored in memory is quicker and more efficient for all uses. In this case, the selected information comprises the feature space pulled out of the overall measurement space. This data is targeted due to its usefulness in building other data from it, such as being able to calculate acceleration at a specific instance in time for some vehicle.

Some general methods for feature extraction have been suggested, such as the "Relief" method discussed by Huang et al. [2004] and Liu et al. [2004]. However those shall not be pursued here, as they are not relevant in light of our intent, based on application domain considerations to use speed and acceleration as features.

Feature selection and extraction methods are often seen in applications with large data repositories. Liu et al. [2004] give examples in relation to genome projects, mining, and market based analysis. These projects use machine learning, or unsupervised learning, algorithms. Once a machine has been trained, it can be fed data to process and dispense results. In this sense, development of a feature space is often performed by unsupervised methods since a trained machine is doing the work.

The resulting set of data is the feature space. Stored in a format conducive to the next step of classification, the data contains the interesting and relevant details needed for the attributes used later. In the case this work, the sought out attributes are velocity and acceleration; thus the feature space contains information needed for those attributes, such as time frame and location for a vehicle at a given time.

Another detail of the feature space is that the information itself can be used for other similar purposes to the work. In this work, for example, the information (as stored from selection and extraction) are also used in creating certain visual representations of vehicles on the roadway.

Classification algorithms take the values from the feature space and parse them, using the classifier, into the classes defined for that feature space. The resulting sets of data should represent all the information originally from the feature space.

The classifier used in this case is *k*-means clustering. The *k* value is often determined by unsupervised learning; alternatively, a specific *k* is defined in this work for use in a supervised approach for clustering the feature space into four descriptive groups corresponding to velocity and acceleration of randomly selected vehicles within the dataset. I consider this a supervised approach due to the a priori determination of the number of cluster groups.

Finally the classes of patterns have been formed. In the case of this thesis, the classes have been predefined for the four traffic types. In other cases, such as unsupervised learning, it will be up to the classification algorithm to group up the data by some similarity. The overall possibilities for classes are numerous for a dataset; in this work, though, it is more useful to focus on four classes that are based on the two key features.

3. MICROSCOPIC TRAFFIC FLOW DATA AND A LIBRARY OF OPERATORS

Section 3.1 is devoted to a description of a general abstract framework for microscopic traffic flow datasets, along with specifications of two publicly available concrete instances of such datasets. Section 3.2 is concerned with a library of operators that I have designed, and implemented in MatLab®, that are intended to facilitate extraction of features from microscopic traffic flow datasets that are organized according to the framework of the preceding section.

3.1 – A General Framework for Microscopic Traffic Flow Datasets

The application creates a feature space from measurements for the purpose of performing cluster formation and analysis. This measurement space exists in the form of traffic datasets. Although the luxury of selecting specific measurements is beyond the grasp of this proposed work, I do have access to datasets that provide a large measurement space to work with. One such dataset, collected by JHK and Associates[2] for U.S. DOT [1985] and reported by Smith [1985], provides an hour of observations, separated by one second, over roughly one-quarter mile long sections of roadway. The data files include time of measurement, vehicle identification, values to represent vehicle type and size (respectively), speed, distance into the measured section of roadway, distance from the right edge of the roadway (when looking down upon the section), a

[2] JHK and Associates is a civil engineering research firm, with multiple offices in the USA, that performs traffic and traffic-related studies and data collection.

value to represent vehicle color, and a lane number. Some of these fields are unnecessary for present purpose, (eg. vehicle type, color, distance from right edge of the roadway, and vehicle size). This observation is an example of feature selection at work.

We conceive a microscopic traffic-flow dataset as organized as follows:

Metadata = <number_of_patterns, time_between_frames, units_for_time, number_of_frames, length_of_section, units_for_lengths, number_of_lanes, graphical_description_of_measured_section, textual_description_of_measured_section, textual_description_of_external_circumstances_during_measurements>,

where a "pattern" refers to a unique <time, vehicle> pair, and

Data=$A(5,1:$ number_of_patterns).

Here the columns of the data matrix *A* contain measurements, as follows:

$A(1,i)$ = frame_index _for_*i*th _pattern <1 : number_of_frames>,
$A(2,i)$= vehicle_id_for_*i*th_pattern <as_assigned_in_source_dataset>,
$A(3,i)$= length_of_vehicle_for_*i*th_pattern <continuous ≥ 0>,
$A(4,i)$= longitudinal_location_of_vehicle_for_*i*th_pattern <continuous $\geq 0, \leq$ length_of_section>,
$A(5,i)$= lane_location_for_*i*th_pattern <1:number_of_lanes>.

For example, values of the metadata parameters for the measurements (dataset), from the Mulholland roadway, used in later sections for experiments are as follows:

number_of_patterns=181142,
time_between_frames=1,
units_for_time='seconds',
number_of_frames=3600,
length_of_section=1341 feet (408.5 m),
units_for_lengths='feet',
number_of_lanes=5,
graphical_description_of_measured_section=as in Fig. 29 of Ref. (10),
textual_description_of_measured_section=
as in first paragraph of quotation in Section II,
textual_description_of_external_circumstances_during_measurements=
as in second paragraph of quotation in Section II.

We term the column index of *A* corresponding to a particular pattern as the *pattern_index* of that pattern. A principal use we make of the matrix data structure is the ability it provides to refer to a particular pattern by its pattern_index. For purposes of implementing the feature-extraction operators described below and carrying out the experiments described in the following sections we reorganized the dataset described above as a MatLab® matrix. This permits more efficient access to individual patterns (e.g., those previously selected as members of a random sample) than the alternative of sequentially searching through the original ASCII file for those patterns corresponding to unique <vehicle_id, frame_index> pairs. Of course this reorganization itself is not without computational cost, and this factor played a role in our choice of one of the JHK datasets, as opposed to other alternatives (see Section 3.1.2).

Further efficiencies, especially in implementation of the feature-extraction operators, can be achieved by appropriate detailed organization of the data matrix (*A*). We begin this discussion by noting that among the numerical data parameters that somehow reflect the size of the data, the meta-attribute number_of_patterns clearly will have the largest value. Thus one clearly wants to minimize the number of searches over all patterns. The parameter number_of_patterns can be expressed in either of the ways

$$\begin{aligned}
&\text{number\_of\_patterns=}\\
&\quad \textstyle\sum_{\text{frame\_indices}} \text{no. of vehicles in frame} = \text{number\_of\_frames}*\overline{\text{vehicles per frame}} =\\
&\quad \textstyle\sum_{\text{vehicle\_indices}} \text{no. of frames vehicle in section} =\\
&\quad (\text{number of vehicles})*\overline{\text{frames per vehicle}},
\end{aligned} \tag{1}$$

where the overbars indicate mean values. For a quarter-mile (.40 km) four-lane section and data acquisition over one hour at a rate of one frame per second, typical values of the four parameters on the right-hand sides of Eq. (1) are

$$\begin{aligned}
&\text{number\_of\_frames=3600,}\\
&\overline{\text{vehicles per frame}} = 40,\\
&\text{number of vehicles=4000,}\\
&\overline{\text{frames per vehicle}} = 20.
\end{aligned}$$

These suggest the advisability of organizing the data so as to minimize the necessity of searches over all frames, or over all vehicles, that might be required. Although the present work employs only velocity and acceleration as features, and their extraction involves only searches for the same vehicle_id locally in time (i.e., at adjacent values of the frame_index), other feature-extraction operations (e.g., spacing between a vehicle and its leader) would involve a search for other vehicle_indices, but still would be local in time. For this reason we recommend, at least for purposes of optimizing extraction of microscopic analogs of macroscopic attributes, organizing the data matrix (A) so that data for patterns contiguous in frame_index are contiguous in storage (pattern_index). Note that NGSIM [2007] currently employs the alternate strategy of organizing the data so that patterns corresponding to the same vehicle are contiguous in storage. Some of the functionality of our operator approach is provided by fields for preceding and following vehicles at each frame; however, we believe the operator approach adopted here is more flexible, in light of the large number of possible attributes.

3.1.1 – The JHK Dataset

The set used within this work was collected in the early 1980s in California and Virginia (specifically the District of Columbia area). Information on certain roadway sections was captured via photograph; the information was then manually entered by looking at the photographs and identifying the vehicles in the section.

The datasets, each for a specific roadway, are made up of a certain section length and an hour's worth of time points. Information is stored on a row-by-row basis, meaning everything important is stored in an individual line. Each line specifies a vehicle (known by the vehicle's ID) and is ordered overall by its timestamp and its distance into the section. Vehicles further into the section are listed first.

An example of a section of the dataset is as follows:

```
9   72 6 30  0   84 14 6  2
9   71 1 17  0   73 34 9  3
9   73 1 19  0   66  1 9  1
9   74 1 16  0   25 48 4  5
10    9 1 19 44 1327 31 4  3
10   12 1 16 52 1305 47 8  5
10   11 1 17 40 1277 17 4  2
10   10 1 18 29 1262  9 9  1
```

Each line of data contains nine different values, which are listed as follows: frame number, vehicle ID, vehicle type code, vehicle length (feet), speed (mph), distance from beginning of section to front of vehicle, distance from right edgeline to middle front of vehicle, vehicle color code, and lane number (right lane = lane 1) [Smith 1985]. Some of this information exists within the dataset because it was part of the process of entering the information and keeping references to the visual images for each time period and section.

Vehicles are given an ID as they enter the section (with the next ID going to the newest vehicle that is furthest into the section). The reader should take note of the zeroes in the speed column listed above. Since the section of roadway was photographed, researchers did not have any information with which to recreate the initial velocity of a vehicle that first entered the section. Because of this, the velocity was set to zero for the first appearance of a vehicle in the section. While it would be possible to come up with some methods to create a hypothetical velocity as a placeholder, I decided to retain the zero value due to the fact that it does not cause any detrimental effect to plots or functions in the code.

3.1.2 – The NGSIM Project

NGSIM, or Next Generation Simulation, is a community effort in association with the Federal Highway Administration (FHWA) to improve simulation in regards to traffic research [NGSIM 2007]. Their work aims to improve simulation tools and research results; thus part of the effort involves the data that is needed for such work.

Data formats for traffic research are, up to the point of NGSIM's forming, variable depending on who is forming the data sets and what applications are using them. Such inconsistency not only makes it hard to apply datasets across applications but also makes it difficult for verification of the data and the experiments using the data without exclusive use of the tools used in the first place. It is then favorable to find common elements of all datasets and combine them into a user-friendly data format that aims to be cross platform and portable.

NGSIM's goal is to meet the demand for such a dataset. As described on their website, their goals for datasets come from the following statements:

> Feedback from NGSIM stakeholders indicated a strong need for robust datasets for validating and enhancing their models. In particular, stakeholders expressed a need for two types of data including:
>
> - Vehicle trajectory data: These are detailed, sub-second vehicle position data which are typically used in simulation models development, estimation and validation. These data are needed to improve simulation modeling efforts.
> - Aggregate data: These are aggregate traffic data, at the same locations as the vehicle trajectory data if possible, over a larger area. These data can be used for simulation model validation and development purposes.[NGSIM 2007]

The datasets follow a similar format to the JHK. The overall sorting is done by time, with a vehicle being tagged with an ID upon entering the section. The ID is part of the sorting as well, since vehicles can overtake one another before exiting the section.

This common formatting for datasets is likely the future of formatting of traffic data. With that in mind, it was looked at as a choice for use in these experiments. Despite the potential for future work complying with this format type, it was rejected from this work for the time being due to its size. Many of these datasets have multiple data points per second, which is beyond the level of complexity desired for this work. The work itself does not prohibit future use of NGSIM datasets, though. The function used to read the JHK files and create the matrix of values could be potentially modified to access NGSIM datasets and, if necessary, be selective in how many lines of data are collected per experimental run.

3.2 – A Library of Feature Extraction Operators

The information specified for extraction from the dataset, into the data matrix, serves two purposes. Firstly, the vehicle ID provides a unique piece of data to distinguish one vehicle from another within the dataset. Secondly, the accompanying values are used in quick computations for creating velocity and acceleration of specified vehicles at specific times. Data fields required for the extraction of these features are:

> *Frame number*: This corresponds directly to a certain period of time, set at one second differentials in this dataset, in which some vehicle appears. This, along with the vehicle's identification number for clarification, is needed in several computations.
>
> *Vehicle ID*: Each vehicle in the dataset was giving a unique identification number for tracking over multiple frames. This guarantees separation of values.
>
> *Longitudinal location*: The vehicle's distance into the measured section. The actual distance is measured in feet for this dataset. This is a crucial value in calculations.
>
> *Lane number*: This is included in the matrix to allow experiments in which patterns are distinguished by lane location.

The initial steps of the extraction function gather these values from the dataset line-by-line and create a matrix array for storage.

The operators are constructed with the intention of nesting and reuse. A notion of levels is used to distinguish them. NEXT and PREVIOUS are designated Level 1 to emphasize their use by all Level 2 operators in computation. Table 1 describes the expected operators by name, execution, and purpose:

Table 1 - Description of operators, with name, execution, and purpose.

Operator Name	**Execution**	**Purpose**
	Level 1 Operators	
NEXT	Returns the position in the matrix of the next (future) instance in which vehicle j shows up.	NEXT searches forward through the matrix to find the next data point of a specified vehicle in the data set.
PREVIOUS	Returns the position in the matrix of the previous (past) instance in which vehicle j shows up.	PREVIOUS searches backwards through the matrix to find the previous data point of a specified vehicle in the data set.
LEADER	Returns the position in the matrix of the vehicle that is leading vehicle j in the same lane during a specific instance of time. Operator PREVIOUS is used.	LEADER traverses the matrix backwards to find the vehicle traveling downstream in the same lane.
FOLLOWER	Returns the position in the matrix of the vehicle that is following vehicle j in the same lane during a specific instance of time. Operator NEXT is used.	FOLLOWER traverses the matrix forwards to find the vehicle traveling upstream in the same lane.

Table 1 Continued

Operator Name	Execution	Purpose
	Level 2 Operators	
ACCELERATION	Returns the calculated acceleration for a vehicle using its previous, current, and next longitudinal positions. Operators NEXT and PREVIOUS are used to find the proper locations.	ACCELERATION uses three locations associated with the specified time frames to calculate the vehicle's acceleration for time t_i.
VELOCITY	Returns the calculated velocity for a vehicle using its previous and next longitudinal positions to determine current velocity. Operators NEXT and PREVIOUS are used to find proper locations.	VELOCITY uses the former and future locations surrounding the current time frame's location to compute a velocity for time t_i.

In order to better understand the organization, I include this description of the VELOCITY operator. The module receives as input a reference to a pattern value in the dataset. In order to calculate the velocity for the vehicle at that particular frame of reference, the PREVIOUS and NEXT operators are used to determine the location of the vehicle on the roadway at the preceding and succeeding interval of time (one second before and one second later). With this method, the needed code to search backwards and forwards in the code for the appropriate instances of that vehicle is separated into a reusable, easily called function that can also be used by other operators; thus these basic pieces of code are separated and called as necessary as opposed to being repeated in code sections as used.

3.3 – Macroscopic Graphic View of the Data

Given a set of traffic data, the application can construct visual images that assist in giving a more understandable view of the progression of traffic through the section of roadway. In this case, there is actually a method already used in the traffic community: trajectory plots. While it is possible to pull information out of these graphs by looking at them, they are truly more interesting to those who are well practiced at it. To those who are not, it is a longer, more arduous task. That is where the three dimensional, top-down space-time-speed plots come in. By using color as the third property, the speed of the vehicles in the section is now much more identifiable at first glance.

3.3.1 – Trajectory plotting

Trajectory plots describe a vehicles' speed by a line traveling from west to east (along the time axis) and from south to north (along the distance axis). While trying to interpret the speed of the vehicles is difficult without a good mathematical background in trajectory plots, several details are noticeable. Take a look at Figure 1 for example. The first details I note when looking at the plot is that, during free flow, traffic is nicely spaced and speeds seem to be even. When the traffic reaches the congested region, though, we see curves (implying deceleration) as well as vehicles becoming more clumped up. Also notable is the breaking up of some lines, implying that lane changing is occurring as traffic tries to react to the slowdown.

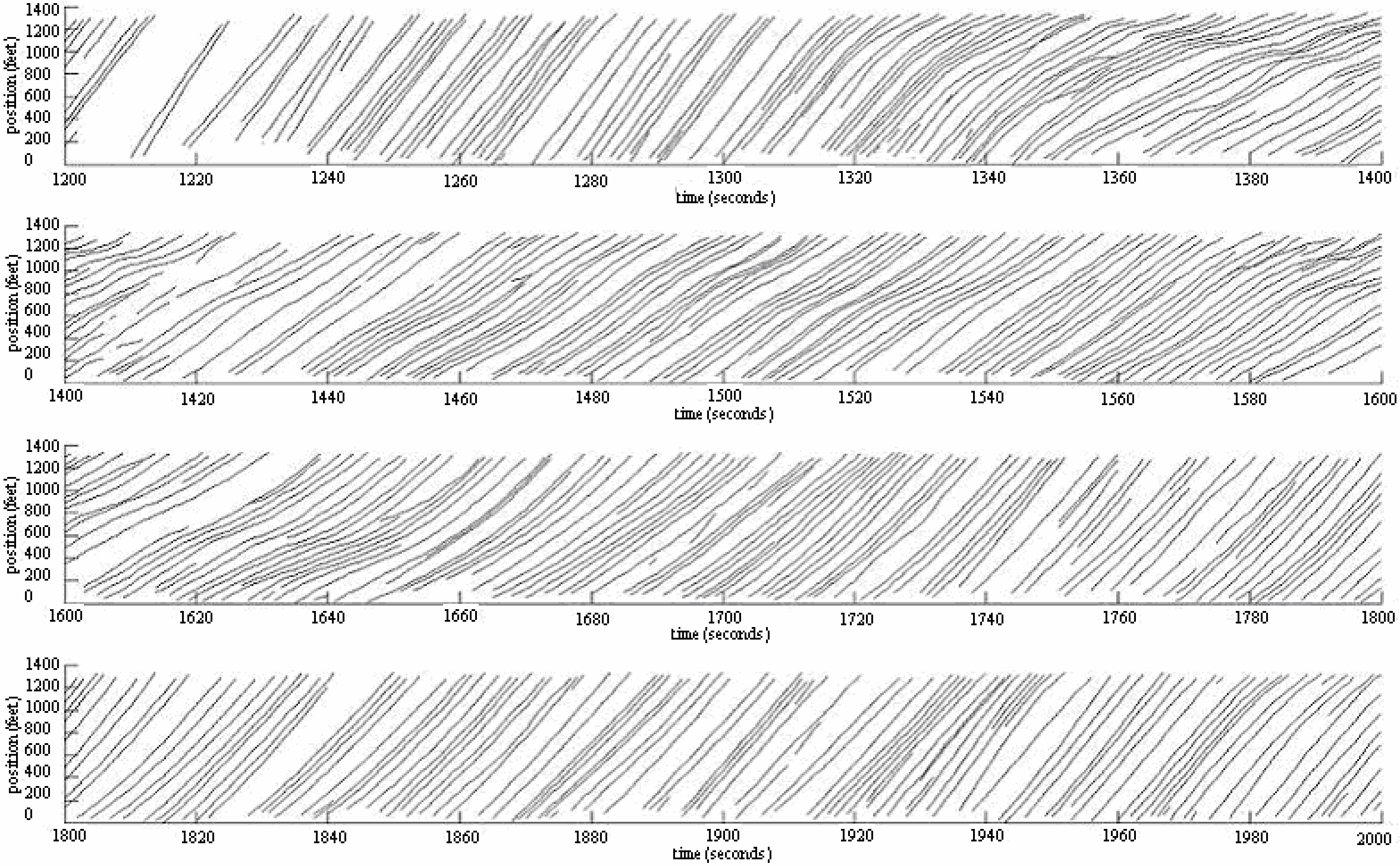

Figure 1 - Example of a trajectory plot.

While this information is enlightening, it falls short. Speeds of vehicles in the section are not readily obvious without relying on pen and paper to calculate it out for individual trajectories. Furthermore, it is difficult to get a sense of how quickly the shock wave propagates upstream. While we are able to see vehicles traveling the beginning of the section at a slower speed than those before the congestion (as well as accelerating at the end of the section in contrast to those early on in the congested region), it does not stand out extremely well without really digging into the numbers.

3.3.2 – Space-time-speed plotting

With such issues in mind, I formed a method for plotting the vehicles' speed on a three-dimensional graph to accentuate the speeds with color. Using a defined colorbar (for values from 0 to 90 feet per second), speeds are more obvious to the viewer just by examining the plot. Looking at Figure 2, we can visibly see free flow speeds in the section by the clear green sections of traffic (we even see some people reaching faster speeds due to the small amounts of blue). At the congested region, though, we notice bands of yellow and red as traffic rapidly decelerates and begins traveling at slower speeds due to the shock wave.

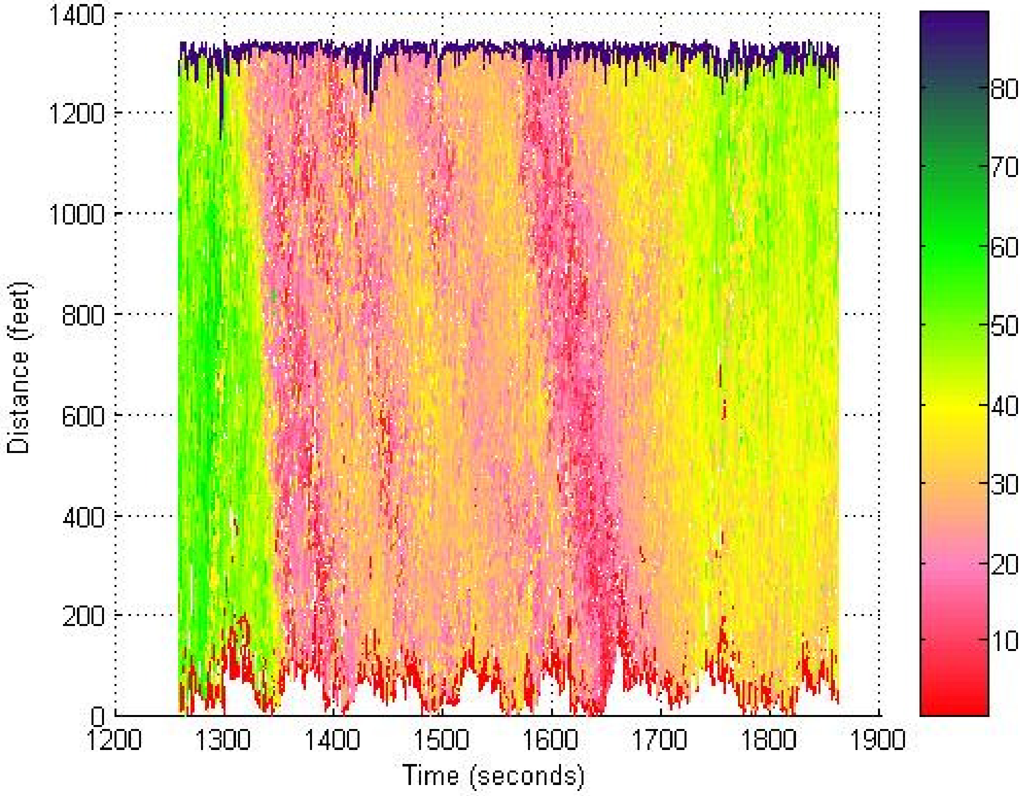

Figure 2 - Space-time-speed plot of time section 1260-1860.

This kind of plot brings the speed of the vehicles clearly to the forefront of the viewers' attention. Rather than having to sit down and individually determine the velocity of vehicles in the section, one can now pull out said information quickly and start trying to read the patterns of the shock waves, acceleration waves, and congested flow.

4. GENERALITIES OF *K*-MEANS CLUSTERING

The following sections elaborate on the *k*-means algorithm (4.1), how and why it was developed (4.2), why convergence is ensured (4.3), its application to this work (4.4), and finally presentation of the visualization plots, silhouettes, which help identify the quality of cluster groups created by *k*-means (4.5).

4.1 – k-means Clustering: The Algorithm

The *k*-means clustering technique is used to classify data into classes, which in this context are often termed as “clusters”. The goal is to divide the patterns (data) into clusters (classes) that provide the best possible fit, in the sense of minimizing the sum of the distance from the individual patterns to the centroids of their assigned clusters. Here “distance” can be defined in the sense of any acceptable measure, but for the present work we consider exclusively weighted Euclidean distances in feature (velocity-acceleration) space.

With the given data, the *k*-means clustering algorithm takes rudimentary centroid values to start the process and then does a recursive two step process to refine those centroid values as well as redistribute any data into other clusters that may be a better fit. A more elaborate description of *k*-means is as follows, with a pseudocode description by Pena et al. [1999] as well as elaboration afterword:

1. Assign data points into *k* clusters (*k* being defined by the user in the present supervised case), where they resemble a cluster mean, in order to be used in forming initial centroid values for each cluster group.
2. At this point, the centroids are calculated and assigned to each cluster group.
3. (Re)evaluate the clusters as follows:
    a. Reallocate data points to their nearest centroid, moving them to the appropriate cluster if it is not their current cluster's centroid.
    b. Recalculate the centroids and the square-error [Pena et al. 1999] of the affected clusters in order to reflect the new cluster mean due to the data point removal/addition.
4. Check if clusters have converged (the square-error of the clusters cannot be further reduced [Pena et al. 1999]), terminating if so. If not, resume at step 3 to move data points until the clusters do converge.

Starting from the first step, the algorithm evaluates the given data and forms *k* initial groups (using a defined *k* value in this work) in order to use in forming the initial centroids in the next step. Once they are determined, the centroids are assigned to the appropriate cluster group and the sum-of-distances for each data point can determined for the next steps.

In step 3, the evaluation (and reevaluation on recursive steps) portion is performed to check if the sum-of-distances for any cluster group is reduced by moving a data point to a different cluster. If a negative result is returned (the sum-of-distances is reduced) by the move, then data point is moved and the re-computation of centroid and square-error [Pena et al. 1999] on the affected clusters occurs.

Finally, a check is performed to determine if the clusters have converged and can no longer reduce their sum-of-distances. If the values have stabilized and benefit no further from moving data points between clusters, the algorithm terminates with the resulting clusters and centroids set.

4.2 – k-means Clustering: Historical Development

The algorithm itself was first introduced by MacQueen as "a process for partitioning an *N*-dimensional population into *k* sets on the basis of a sample" [MacQueen 1967]. MacQueen was working on optimal classification problems; one of the problems boiled down to reduction to a minimized partition structure. A method, *k*-means, was designed to specifically deal with the computation to find the partition that would form the optimal results from the minimized partitions [MacQueen 1967]. MacQueen is quick to mention that it does not find the optimal answer most of the time; in fact, he only presents two examples where it does. Thus the solution he presents does not provide optimality, yet it does "give partitions which are reasonably efficient in the sense of within-class variance" [MacQueen 1967].

One of the details about *k*-means that MacQueen emphasizes is that the procedure is "easily programmed and computationally economical" [MacQueen 1967], making it favorable for use on computers (a fact more considerable then than now, although just as useful in the present) which furthermore makes it favorable for use on large sample sets [MacQueen 1967]. As he states later on in Section 3 of his paper, “Perhaps the most obvious application of the *k*-means process is to the problem of ‘similarity grouping’ or ‘clustering’” [MacQueen 1967]. The algorithm is an assistant to the user due to its method of grouping information by similarities so that pertinent information is clearer. With that in mind, MacQueen emphasizes using the method along with “theory and intuition” [MacQueen 1967].

The *k*-means algorithm needs the user define a value for *k* before running in order to ensure that mean values will be formed for each cluster. As just mentioned, it is appropriate for the user of *k*-means to be familiar with the data being experimented on, but in some situations this may be considered a disadvantage. Intimate knowledge of the data is not always a given; thus if one is looking to experiment to determine a more appropriate number of clusters, there may be limited flexibility in using *k*-means. Furthermore, it is said that *k*-means can suffer due to outlying data points [Fu et al. 1970].

In the case of this work, the number of desired clusters is already known. They were determined based on the traffic conditions I am looking to monitor; as for outlying data points, the hope is that the sheer volume of data points per cluster will prevent any strange values from harming the overall cluster results. There is no guarantee of this in that outliers may end up being a noticeable problem in attempting to get good cluster separation. However, the silhouette index and associated plot introduced in Section 4.5 below provides a means of determining if this problem exists in any given instance.

4.3 – k-means Clustering: Convergence

As described earlier, the *k*-means process involves computing the sum-of-distances from the current centroids for each cluster. This computation, combined with reassignment of data points to closer centroids and re-computation, is done iteratively until a minimum sum-of-distances is reached for all clusters. This does not imply that every execution of a *k*-means will result in the same clusters with the same minimized sum-of-distances each time, but rather that on any given run the expectation is to get the local best fit clusters (and judging them best fit based on that sum-of-distances value). Thus the optimal partitioning into clusters is not always guaranteed, but the resulting clusters should resemble a reasonable assignment to classes.

The discussion of convergence in relation to *k*-means is immediately associated with the idea of having a finite number of iterations. The execution of *k*-means is useless without reaching an end point (and thus clearly defined clusters) but can we ensure that it will end? As discussed in [Selim and Ismail 1984], convergence to a partial local minimum will occur within a finite number of iterations. As the authors emphasize, this is not guaranteed to be the global minimum for the clusters, but it does guarantee an eventual result to an execution of *k*-means on a set of data.

4.4 – Use of k-means Clustering in This Application

A sample of 1000 patterns was used to create a "training set" for 4-means clustering to implement in the experiments described in Section 5. A seed, set specifically rather than using the system clock, was fed into a random number generator; the resulting set of numbers was then used to select 1000 random patterns (lines of the data matrix). The reason for setting the seed manually is to allow for trying multiple seeds in the process (as will be seen in the experiment setup and results in Section 5). The goal was to get an initial set of centroids to use with the rest of the experimentation as a base case (except when modified for purposes of specific experiments).

From the 1000 selected patterns in the matrix, the velocity and acceleration is computed for each data point. Figure 2 in Section 3.1.2 is actually the space-time-speed plot from this run. Using the operators discussed in Section 3, the preceding and succeeding data in the matrix for that selected pattern are used in these computations. The velocity and acceleration values used in the process of determining the clusters were normalized during computation, using the variables *vnorm* and *anorm*, respectively.

The velocity values, in all cases, were divided by 90, to produce normalized values having a maximum on the order of one. With *vnorm* so set, experiments were run to focus in on a good *anorm* value. Experiments were run around *anorm = vnorm/2,* with lower values (10, 15, and 25 feet per second per second), mid-ranged values around the halved *vnorm* (45, 50, 60), and even with higher values (100, 1000). to see where acceleration was given enough weight. In the end, a value slightly above half of *vnorm* was selected, specifically assigning *anorm* to be 50. Thus all acceleration values were divided by 50. The goal was to minimize variation in acceleration changes that may exist from bad, outlying data.

With the values now normalized, the resulting information from the patterns is fed into Matlab®'s *k*-means function. The 'Distance' parameter is set to 'sqEuclidean' and ‘display’ is set to ‘iter’. A value of 4 is also fed into the function to specify four clusters to be created from the input. Finally, a set of initial centroids is given to *k*-means as a start setting. Initializations for clusters 1-4 are as follows, respectively: [0/*vnorm*, 0/*anorm*; 50/*vnorm*, -10/*anorm*; 50/*vnorm*, 5/*anorm*; 100/*vnorm*, 0/*anorm*]. The *k*-means function gives a set of resulting centroids as well as output that can be used in a silhouette plot, including the cluster to which each pattern is assigned.

Table 2 lists the centroid values created from this training case. Notice that the shock and acceleration waves show a more pronounced deceleration and acceleration than congested and free flow; this is to be expected since we should see more stable speeds for the vehicles in the latter two clusters.

Table 2 - Values of centroids for the four clusters from 1000 point random sample.

Cluster number (and name)	X-axis value (Speed, feet per second)	Y-axis value (Acceleration, feet per second per second)
Cluster 1 (Congested flow)	6.7841	-0.6742
Cluster 2 (Shock wave)	55.4545	-4.0553
Cluster 3 (Acceleration wave)	42.1271	3.3952
Cluster 4 (Free flow)	76.6460	1.3292

The silhouette plot (Figure 3) created from this training set has a mean silhouette value of 0.5694, as compared to an ideal of 1 and a worse case of -1. The function “silh” within Matlab® is used to produce the plot as well as return the mean distance. The explanation of this “mean distance” and our interest in silhouette plots for *k*-means clustering is in Section 4.5. Without better knowledge on understanding silhouettes in this matter, though, one can view the plot initially and possibly gather the concept that the closer to 1 the values for a cluster get, the better the fit.

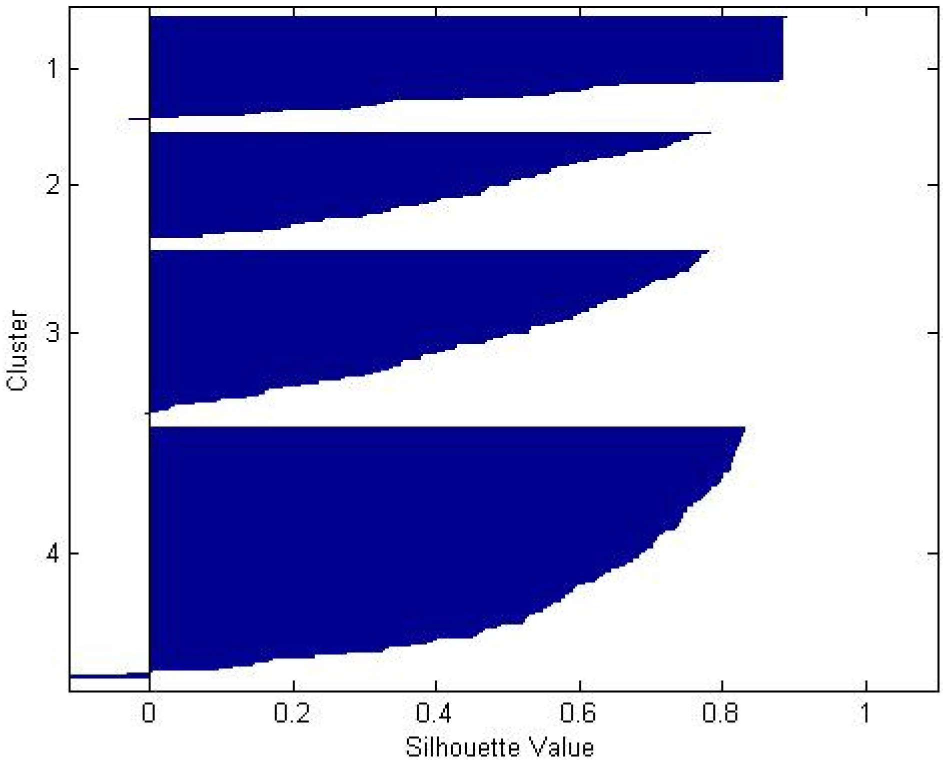

Figure 3 - Silhouette plot of clusters from centroid creation using 1000 random samples.

4.5 – Silhouette Plots

As previously discussed, *k*-means does not always give an optimal solution; in fact, it does only in specific cases. Thus a measure of the quality of fit of clusters is appropriate. One such method is silhouette plotting, as discussed in [Rousseeuw 1987]; here we have a visual tool for examining fit.

Given a pattern and a clustering (set of clusters), the mean distance from that pattern to all of the patterns in a particular cluster will be termed as the distance between that pattern and the specific cluster. The unnormalized silhouette value of that pattern, relative to that clustering, is the distance from that pattern to the closest cluster (one at the smallest distance) except for its own cluster minus its distance from its own cluster. As described on the Matlab support website,

> The silhouette value for each point is a measure of how similar that point is to points in its own cluster compared to points in other clusters, and ranges from -1 to +1. It is defined as
>
> `S(i) = (min(b(i,:),2) - a(i)) ./ max(a(i),min(b(i,:),2))`
>
> where `a(i)` is the average distance from the `i`th point to the other points in its cluster, and `b(i,k)` is the average distance from the `i`th point to points in another cluster `k`. [The MathWorks, Inc. 2007]

Ideally a clustering would assign each pattern to a cluster at a much smaller distance than any other cluster. In such a case the corresponding silhouette value would be near one. At the other extreme a pattern conceivably could be assigned to a cluster at a much larger distance than that of the cluster nearest it. In such a case (of gross misassignment) the corresponding silhouette value would be near -1. Between these two extremes a value near zero indicates a pattern that is difficult to assign to a cluster, and a value greater than 0.5 suggests a point that is likely assigned to the “right” cluster.

The best single figure of merit for a particular clustering is probably its “silhouette index,” which is defined simply as the mean silhouette value, over all patterns. The larger this value, the more sharply the clustering segments of particular patterns. A more detailed graphical display of the quality of a particular clustering is provided by its silhouette plot. In a silhouette plot, a segment of the vertical axis is assigned to each cluster, the patterns within each cluster are ordered from largest (top) to smallest (bottom) silhouette value, and a horizontal bar, of length proportional to the corresponding silhouette index, is drawn for each pattern.

To further analyze the quality of said resulting centroids, the silhouette index of the resulting silhouette is recorded. This value, as calculated from all patterns, gives an idea of fit since it gives an overall perspective of how well partitioned the clusters are in a single, average value over all four clusters. A higher mean implies a better overall fit for the four clusters.

5. APPLICATION TO MICROSCOPIC TRAFFIC DATA

This Section describes the four different cases that will be run, with discussion on specific interesting variables per case. Each of these cases, other than the reference case, exist for one of two reasons: 1) They apply the training set centroids to a different time section of interest to make sure details are visible, or 2) they either create a specific set of new centroids to apply to the 1280-1860 time section from the Mulholland dataset.

The reference case, using the 1000 sample training set centroids from the Section 4.4, is 5.1. This same centroid set is being used on a later section of roadway in Section 5.2. An experiment increasing the training set to 2000 patterns is the focus of 5.3. Finally, the random seeds used in the random selection of the training set are modified for the case in 5.4.

For each subsection, an elaboration on choices is made; furthermore, cluster values and silhouette plot are given for each modified training set execution. The results for each case's application to the selected time-slice are shown in their respective place within Section 6, unless otherwise noted.

5.1 – Reference Case

As mentioned, the centroids created from the 1000 sample training set in Section 4.4 are applied for this case. Within the Mulholland dataset, there are two incidents found in the data that show an adverse effect on the flow of traffic. The first incident is found in the section 1260-1860. A visual representation of the incident time slice, created to identify the expected KWM flow of traffic in response to the incident, is given in Figure 4. The expectation in the following plots is to see a representation of the shock wave, a region of congested flow, an acceleration wave that propagates upstream faster than the shock wave, and the resulting forward shock after the acceleration wave catches up to the shock wave.

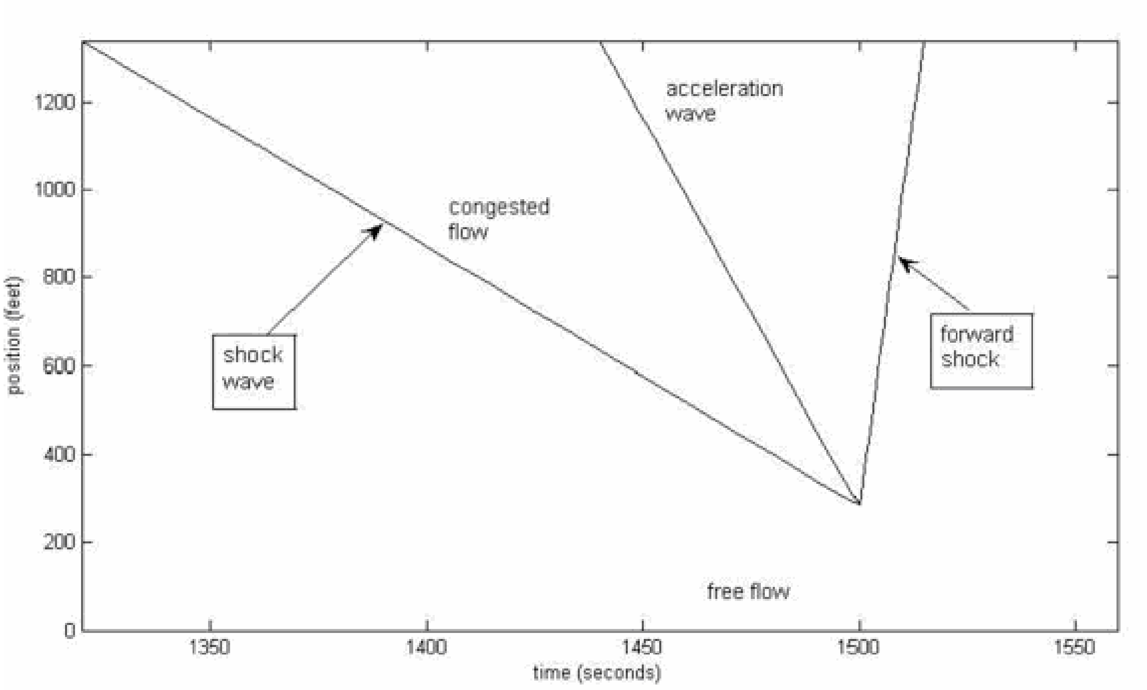

Figure 4 - A representation of the incident time as expected from KWM.

The focus of the reference case is this time section; the intention is to see how well the centroids work when data is divided into four clusters based on the centroids created in Section 4.4. The space-time-cluster plot (which shows data as partitioned into the four classes) as well as the silhouette plot to determine fit are shown in 6.1. The following three subsections will follow the same format in terms of displaying their respective plots.

The second listed incident in the Mulholland dataset is found in the time section 2060-2460, and is used in the experiments recorded in Section 6.4. Once again, the centroids from the reference case will be applied to see if results and plots are consistent in revealing details about the flow of traffic. It should be noted that this is the only case that uses this time section. The other three cases all apply their centroids to the 1260-1860 time section.

5.2 – Increased Random Sample Size

With two experiments done with the 1000 sample size, the next step is to see if doubling the sample size to 2000 random patterns creates an improved set of centroids. The centroids for this run are given in Table 3 below.

Table 3 - Values of centroids for the four clusters from 2000 random pattern sample.

Cluster number (and name)	X-axis value (Speed, feet per second)	Y-axis value (Acceleration, feet per second per second)
Cluster 1 (Congested flow)	4.5883	-0.1090
Cluster 2 (Shock wave)	57.6441	-3.1111
Cluster 3 (Acceleration wave)	40.8177	2.1204
Cluster 4 (Free flow)	76.9050	1.6185

These new values do not show much separation from the previous centroids values in the 1000 sample. The most noticeable difference is in cluster 1, where the speed value is reduced by roughly 2 feet per second and the acceleration value is reduced by .5 feet per second per second. In both cluster 2 and 3, the speed and acceleration both see slight reductions. Finally in cluster 4, both the speed and acceleration see slight increases.

A silhouette plot was produced to show the fit for the centroids of the reference training set case of Section 4.4 (Figure 3). Likewise, Figure 5 is the plot of the silhouette created from the 2000 pattern experiment, with a mean silhouette value of 0.5302.

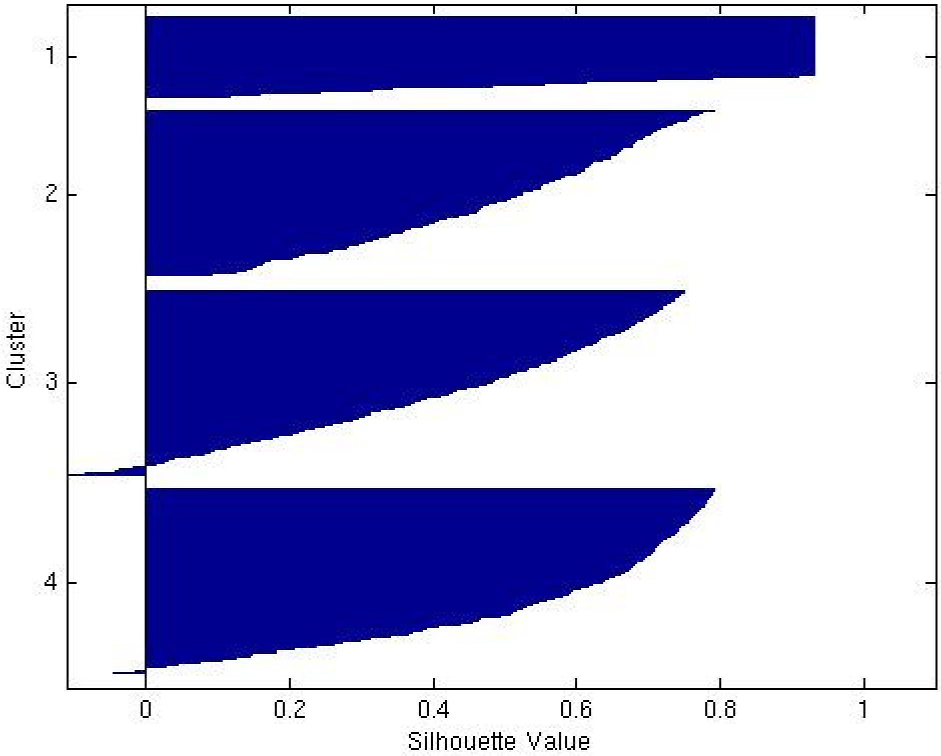

Figure 5 - Silhouette plot of clusters from centroid creation using 2000 random samples.

The plot reflects an increase in the relative number of patterns for cluster 2, which picks up the most new patterns from the 1000 additional patterns. Cluster 3 also now reflects some misclassified values that it did not show in Figure 3. Aside from these differences the results are similar to those of the base case of Figure 3.

5.3 – Variations of Random Seeds

The final experiment reverts back to the 1000 random sample patterns; the twist in this case is that the seeds for the random number generator are changed to see how much of an effect differently selected random patterns have on the centroid creation process. In this case, two new seeds were used.

Table 4 - Values of centroids for the four clusters from 1000 point random sample, first new seed.

Cluster number (and name)	X-axis value (Speed, feet per second)	Y-axis value (Acceleration, feet per second per second)
Cluster 1 (Congested flow)	5.4874	-0.4202
Cluster 2 (Shock wave)	52.4864	-5.0090
Cluster 3 (Acceleration wave)	42.3993	3.2085
Cluster 4 (Free flow)	74.8630	0.8697

The changes versus the Table 2 are fairly small, with a noticeable reduction in the speed as well as an increased negative acceleration for cluster 2. Cluster 4's acceleration value is reduced by nearly .5 feet per second per second as well as taking a small reduction in speed.

The centroids created from each new seed are listed in Tables 4 and 5, respectively, along with their corresponding silhouette plots, Figures 6 and 7.

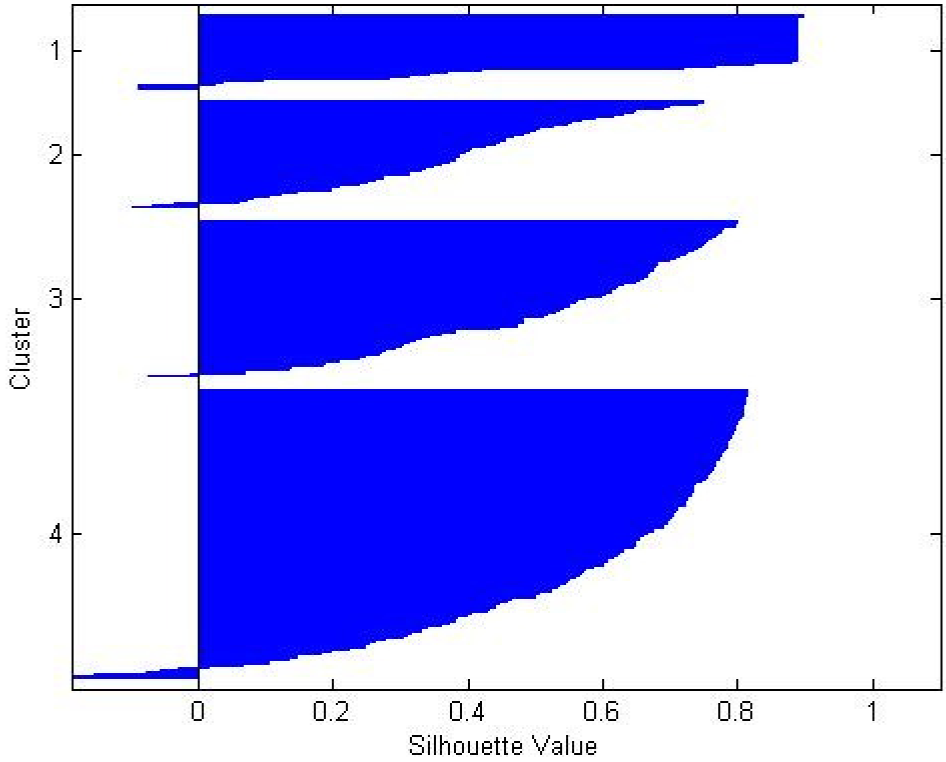

Figure 6 - Silhouette plot of clusters from centroid creation using first new seed.

This need seed apparently leads to an inflation of values in cluster 4, as it is larger than when we first see it in Figure 3. The appearance of negative values in all four clusters shows some misclassification over all four clusters whereas it was limited to clusters 1 and 4 in the initial training set.

Table 5 - Values of centroids for the four clusters from 1000 point random sample, second new seed.

Cluster number (and name)	X-axis value (Speed, feet per second)	Y-axis value (Acceleration, feet per second per second)
Cluster 1 (Congested flow)	4.4057	0.0410
Cluster 2 (Shock wave)	49.9093	-4.1032
Cluster 3 (Acceleration wave)	46.1487	4.7198
Cluster 4 (Free flow)	74.7881	0.2465

An immediately identifiable change in this centroid set is the positive acceleration listed for cluster 1. This is the first occurrence so far of a non-negative value for that case. Three values take notable losses, specifically the speed values for cluster 2 (which drops 5.5 feet per second) and cluster 4, as well as another large drop in the acceleration value for cluster 4. Cluster 3 sees an increase in both speed and acceleration, by roughly 4 feet per second and 1.3 feet per second per second respectively.

The silhouette plot for this new seed once again shows cluster 4 having more patterns than the rest. This time, though, the number of clusters with possible misclassified patterns is down to two clusters again (3 and 4).

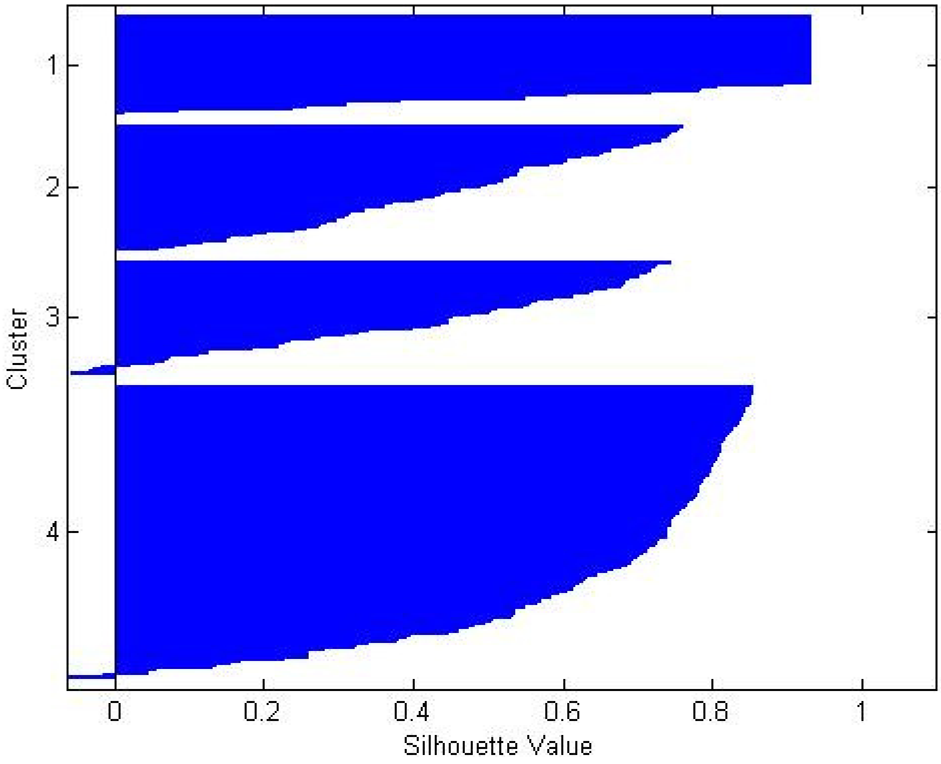

Figure 7 - Silhouette plot of clusters from centroid creation using second new seed.

6. RESULTS FOR CLUSTERING USING CENTROIDS

Section 6 gives the resulting plots for the cases described in Section 5, maintaining the section structure with the reference case as 6.1, the increased random sample size as 6.2, the varying random seeds as 6.3, and finally the later time section as 6.4.

In all of these cases, the procedure is as follows. For all patterns within the time section, loop through and assign each pattern to their nearest centroid. The process for determining the minimal distance is to take the pattern's velocity and acceleration and find its distance to the first cluster's centroid. Then, the distance to the centroid of the other three clusters is computed in succession. If the pattern is closer to the centroid of another cluster, it is assigned to it and once again the minimal distance measurement is iterated to make sure it is in the right cluster with its nearest centroid. The results are then plotted in a space-time-cluster plot as well as an associated silhouette plot.

The space-time-cluster plot is purposefully colored and presented in a fashion similar to the space-time-speed plot shown earlier. The x-axis represents the time frame while the y-axis represents the distance into the section. The aim in selecting the four colors for the cluster is to create a plot that will be visually comparable to the space-time-speed plot; thus the space-time-cluster plot's results can be analyzed to see if it gives a good representation of the time section.

6.1 – Reference Case

The centroids from the reference case were applied to the 1260-1860 time section of data to produce the space-time-cluster plot seen in Figure 8. This is a visualization of the four clusters plotted in a similar format to the space-time-speed; the idea is to see if the clusters produce an improved ability, relative to the earlier space-time-speed plot of Figure 2, to visualize the occurrence of the different kinematic-wave classes of flow.

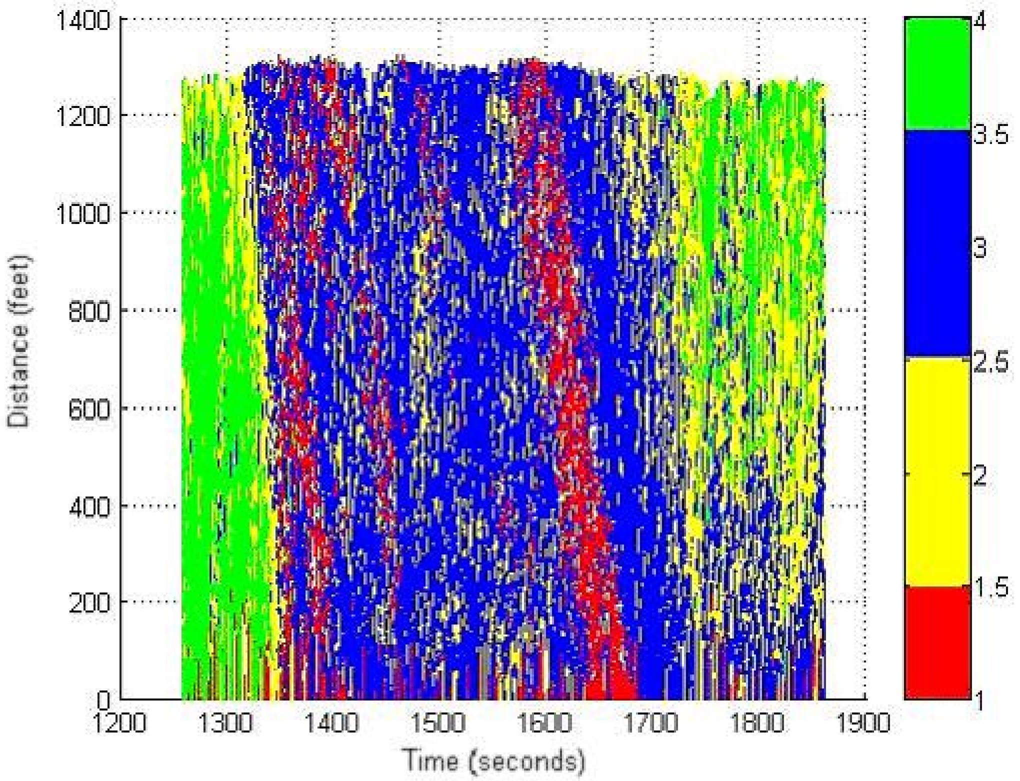

Figure 8 - Space-time-cluster plot of time section 1260-1860, the reference case.

The plot itself shows an initial resemblance to the space-time-speed plot for the time frame. Leading up to the initial shock section there is obvious free flow speed, followed by some noticeable decelerating vehicles (who presumably saw the shock wave approaching and reacted). There are seen as a solid yellow band (shock wave) propagated upstream (starting around 1320), followed by a wide section of congested flow and varying acceleration (blue), deceleration (shock, yellow) waves, and congested regions (red). As was seen in the space-time-speed plot, the traffic improves (returns toward free flow) near the end of the time frame; here a block of free flow speed is seen entering from downstream while the upstream traffic continues to correct itself to join the free flow group.

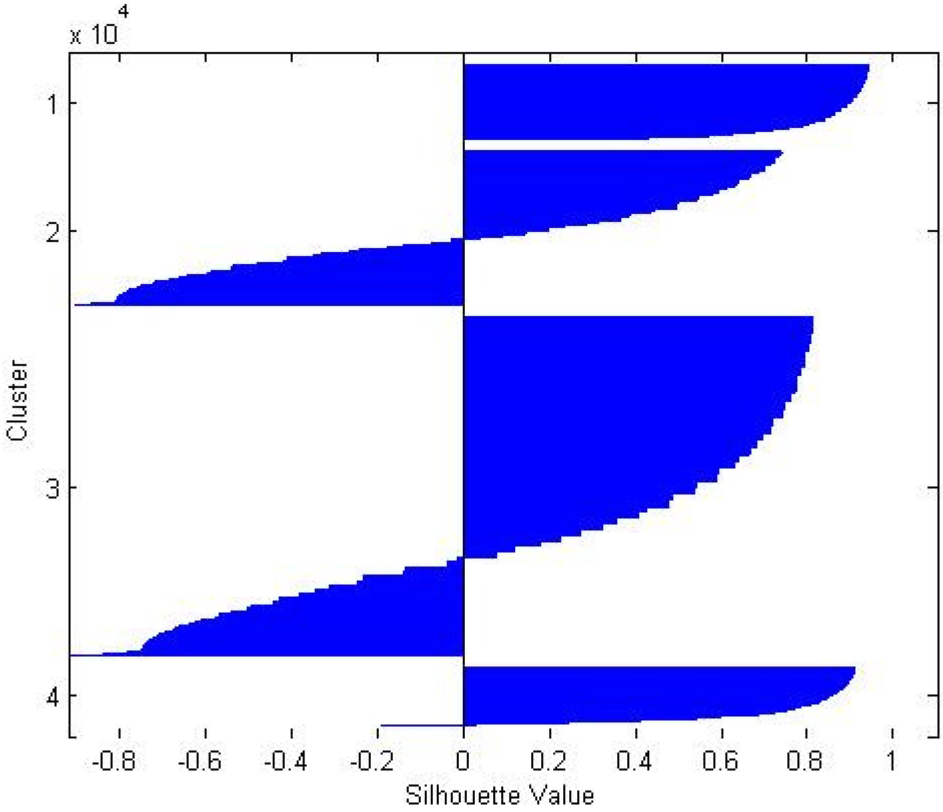

Figure 9 - Silhouette plot of time section 1260-1860, the reference case.

Figure 9 is the silhouette plot of this cluster group for 1260-1860 with the given initial cluster values. The mean value over all four clusters is 0.3455. The total number of vehicles for clusters 1 through 4 ended up as 4439, 9082, 19960, and 3428, respectively (these values are included for comparison to other cluster plots in later sections).

Both the shock wave and acceleration wave clusters have an abudance of misclassified values. The reasons for these two clusters being so poorly represented are not obvious, but a possible answer lies in the method for obtaining the training set. Since the random selection is done over the entire data set, there is no attempt to focus on certain types of traffic. Instead, the selection assumes that patterns are equal over the course of the entire hour long time slice.

In the time slice of interest observed in this Section 6, though, we are looking at traffic that, at times, is in a very congested state. Thus there are a large number of patterns that reflect a lower velocity than is seen over the course of the hour long section. Since the clustering result is being performed on a specific time slice, it is worth examining in future work whether or not the training set should take from the entire data set or from a sample that more accurately reflects the time slice. If so, the distance function as well as normalization values would need reconsideration as to their appropriateness for the training set being used.

6.2 – Increased Random Sample Size

The centroids created from the 2000 random pattern sample produce the space-time-cluster plot seen in Figure 10.

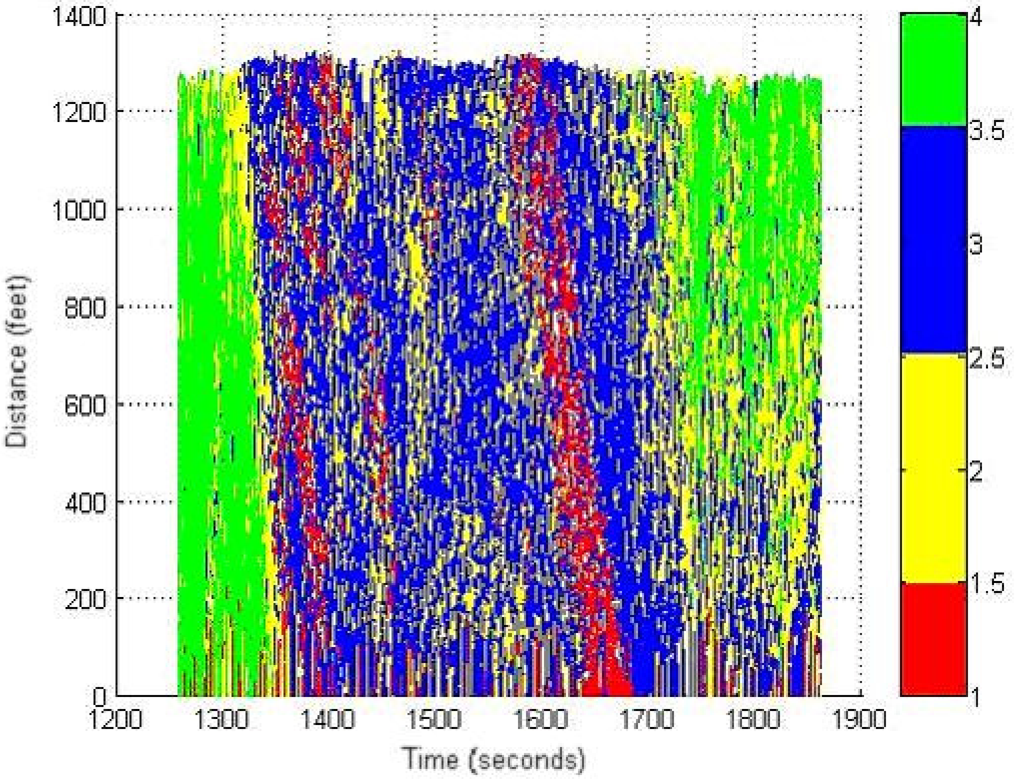

Figure 10 - Space-time-cluster plot of time section 1260-1860, using 2000 random sample.

It is noticeable that less traffic is identified as congested flow versus previous cluster plots. Whereas the reference case shows a thicker band of red at the initial slowdown leading into the trouble section, this centroid group places more vehicles into the deceleration wave cluster.

Figure 11 is the silhouette plot for this example, with a mean value of 0.2461 over all clusters. The number of vehicles for clusters 1-4 is 3519, 11640, 16975, and 4775, respectively.

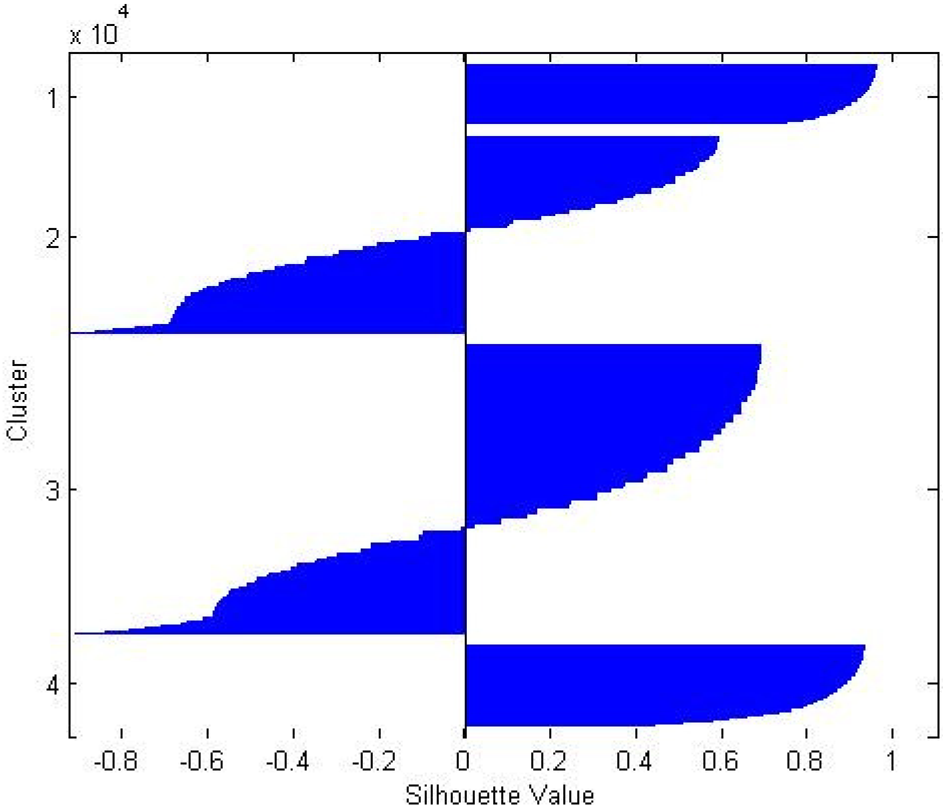

Figure 11 - Silhouette plot of time section 1260-1860, using 2000 random sample.

Once again the acceleration and deceleration waves show some disappointment, more so in this case since the positive values are not strong enough to match the previous cases

6.3 – Variations of Random Seeds

The first new seed's centroid values produce Figure 12 when inserted into the 1260-1860 time section. Since the cluster plot looks similar, the silhouette plot in Figure 13 presents the interesting detail of this seed's performance.

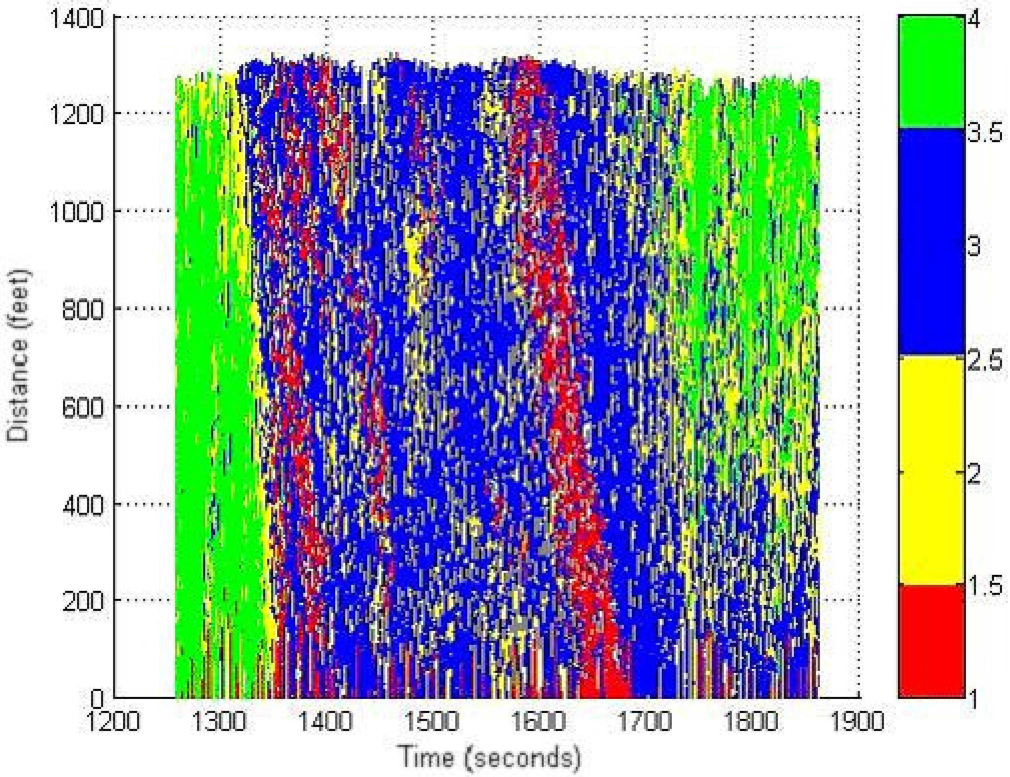

Figure 12 - Space-time-cluster plot of time section 1200-2200, first new seed.

The mean value over all clusters is 0.2565; this is a fairly large loss of accuracy in clusters versus the reference case. The number of vehicles in clusters 1-4 is 4123, 9237, 19053, and 4496, respectively. Despite the cluster plot seeming familiar, the silhouette plot shows that this seed's selected patterns did not form a more robust centroid set.

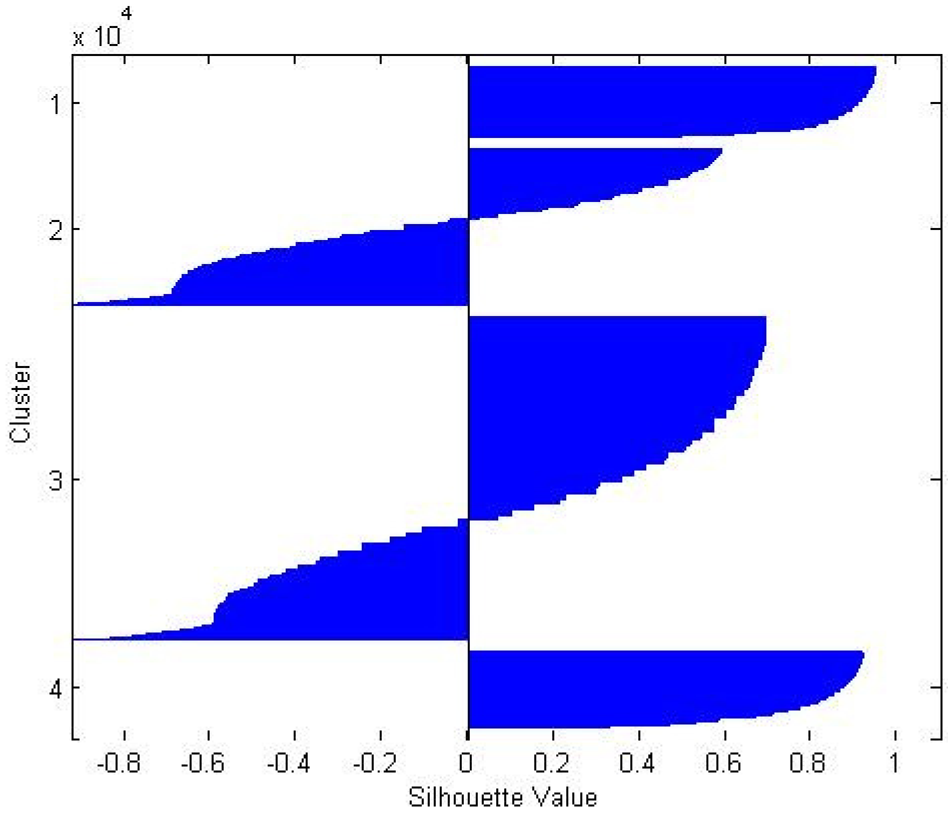

Figure 13 - Silhouette plot of time section 1200-2200, first new seed.

The results for the second seed once again show a recognizable cluster plot (Figure 14) but a poor silhouette plot (Figure 15). An interesting result appears in the number of vehicles for each cluster. In this case, the shock wave cluster has just under 3000 more vehicles present than the reference case while the acceleration wave has 4000 less vehicles present. Despite this reduction, the number of misclassified patterns is still large. The number of vehicles for clusters 1-4 is 4482, 12786, 15295, and 4346.

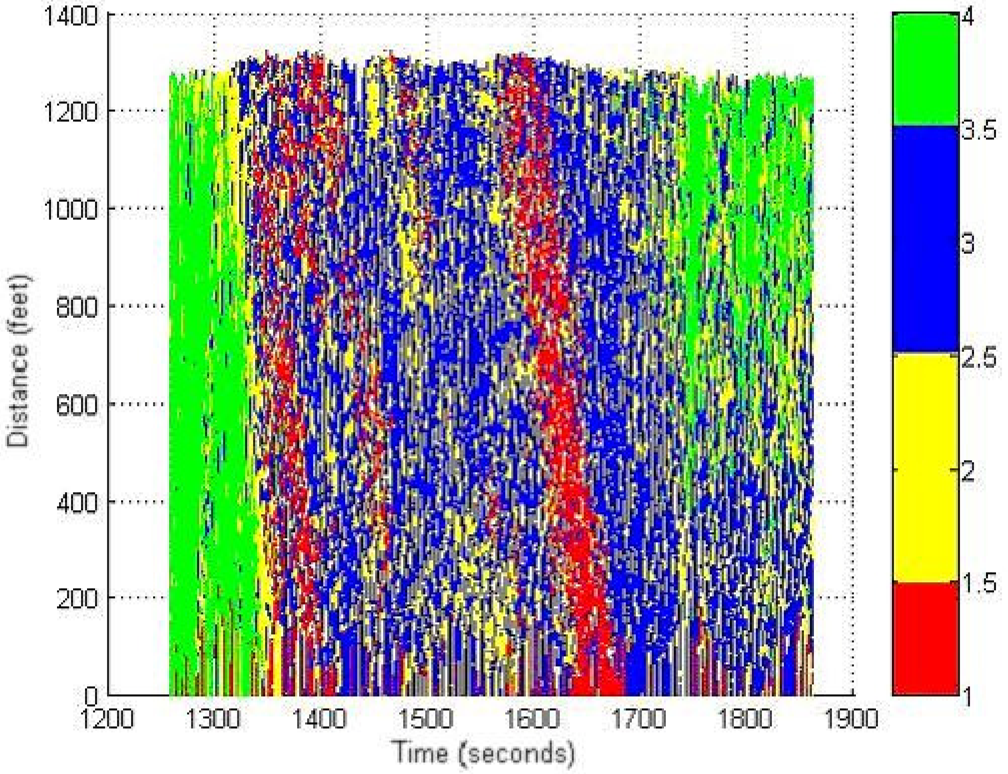

Figure 14 - Space-time-cluster plot of time section 1200-2200, second new seed.

Despite these differences, the silhouette for the deceleration and acceleration wave clusters shows a lot of misclassified values. The mean over all clusters is 0.1411. It should be noted that despite the fact that two changes resulted in degraded performance, these changes were not based on any specific reasoning, just seeds. In other words, just as changing the seeds can produce less reliable results, I would venture to guess that another seed change could potentially improve the performance.

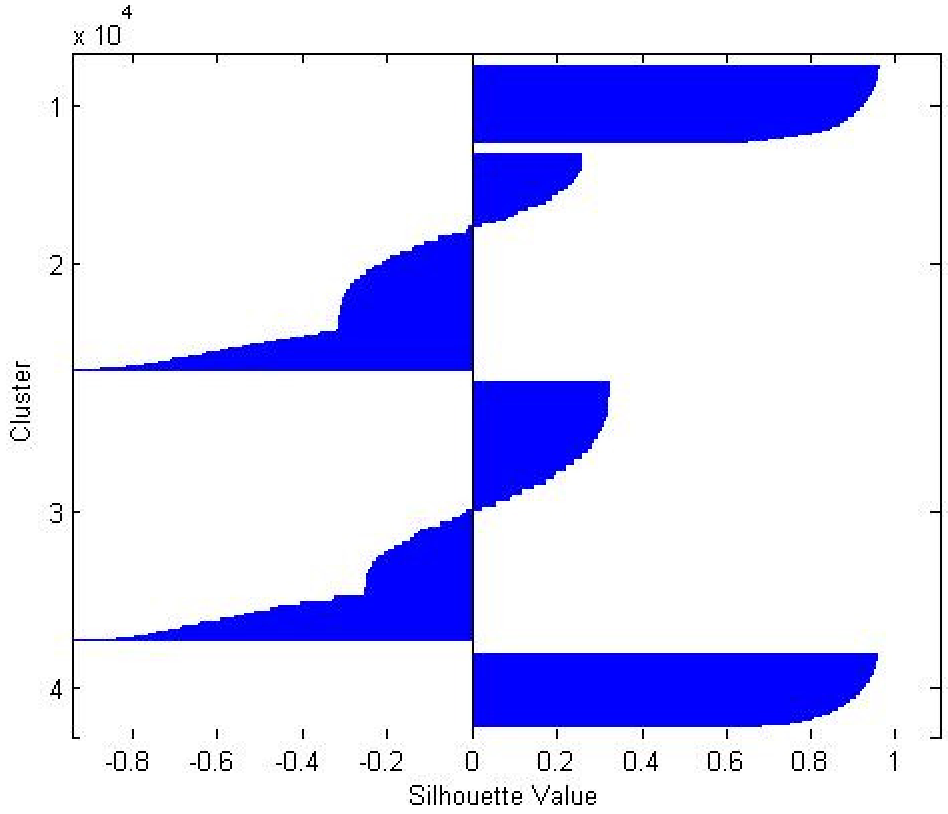

Figure 15 - Silhouette plot of time section 1200-2200, second new seed.

Guessing seeds would not be a productive or scientific method for improving this; that being said, unless the first seed used was surprisingly the best available (unlikely), there is at least the possibility to find better results if only be figuring out a better process for getting the random pattern samples.

6.4 – Later Time Section from Dataset

The 2060-2460 time section's clusters are shown in the time-speed-cluster plot in Figure 16. Although the section shows very little congested flow, there might be signs of a truck slowing down traffic (this is the suggested explanation for the slow down as given by [25], listed the event at 2280 [38 minutes]). From roughly 2170 and on, you see a group of acceleration and deceleration waves apparent in the middle of the section and downstream. Furthermore around 2300 there is a small group of deceleration wave vehicles near the downstream edge of the section. Both these groups could be showing the effect of the truck on the overall traffic going through the section; thus perhaps the clustering is showing traffic as it approaches the truck and attempts to pass it. The lane changes could be the explanation of slowing down; likewise, passing and reentering the truck's lane downstream could be the explanation of the speeding up.

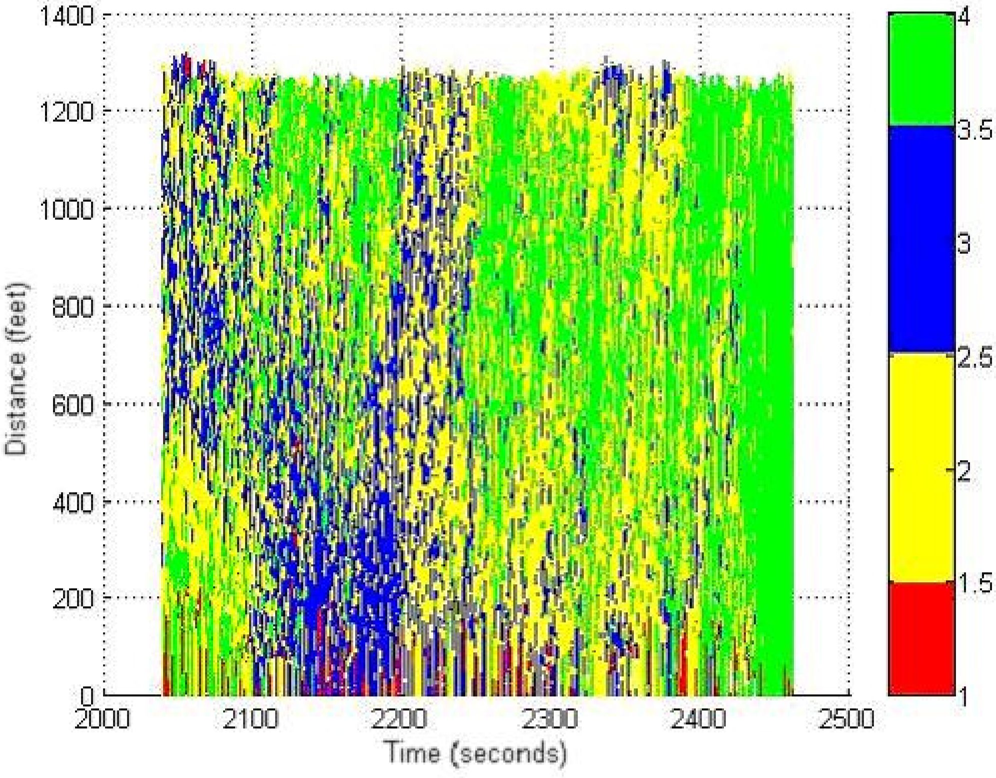

Figure 16 - Space-time-cluster plot of the 2060-2460 section.

The silhouette plot for this 2060-2460 section is shown in Figure 17, with a mean of 0.3158 over all clusters. The number of vehicles in clusters 1-4 is 24, 8642, 3819, and 6238, respectively.

The congested flow cluster shows little in terms of results due to the lack of stalled traffic in this time section, and is thus not interesting in this example. Once again the acceleration and deceleration wave clusters show some weakness, although in this example they tradeoff as the deceleration wave shows some improvement while the acceleration wave degrades. The free flow cluster remains fairly respectable despite some negative values that perhaps would be better assigned in the lacking congested flow cluster.

The examples from these training sets show that the *k*-means implementation has not identified the kind of results expected to reflect well produced clusters. While there may be solutions to obtaining more accurate results from this methodology, Section 7 examines another technique for dividing the patterns into clusters.

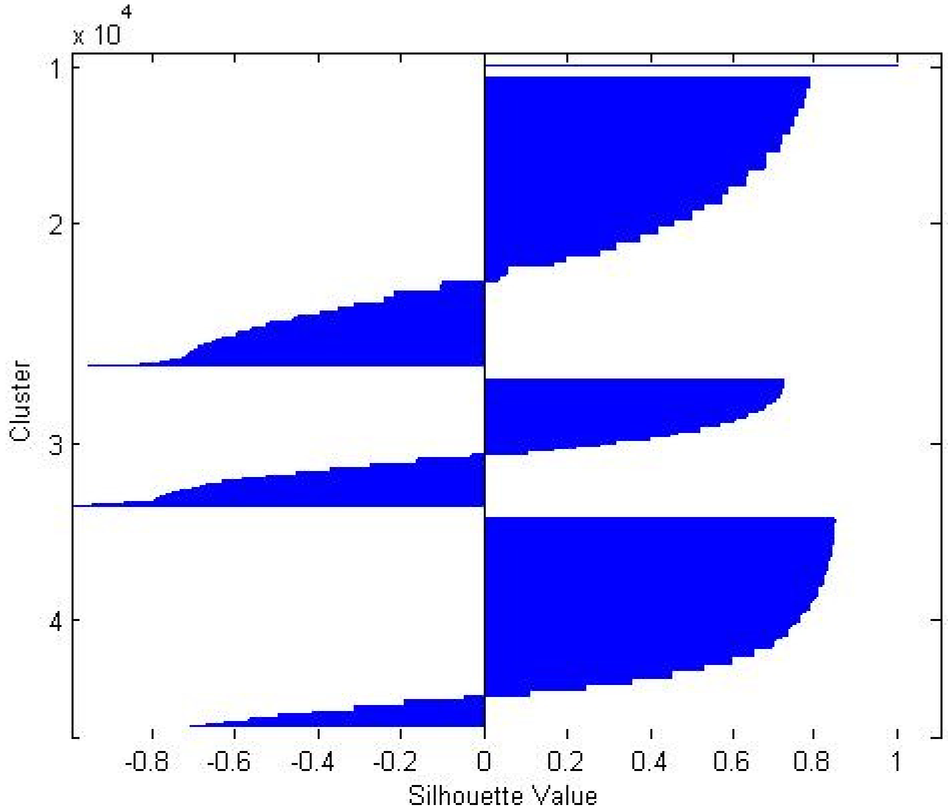

Figure 17 - Silhouette plot of time section 2200-3200.

7. CLUSTERING FROM DECISION TREES

Another method considered for experimentation is the idea of creating clusters using a decision tree. Here, each pattern is assigned into one of four clusters based on where it fits in using certain values to distinguish for velocity and acceleration/deceleration. An individual data point is assigned first by whether or not its acceleration is less than some deceleration threshold or greater than some acceleration threshold, in which case it is assigned respectively to the shock-wave or acceleration-wave cluster. Assuming it falls in between, it is assigned to the congested or free-flow cluster depending on whether its velocity is below or above some velocity threshold. The original values of velocity and acceleration are used in the decision tree as opposed to the normalized values used for the 4-means clustering.

Section titles specify the thresholds used by the decision tree. So, in order, they define the deceleration threshold, the acceleration threshold, and the velocity threshold. A wider time slice (1200-2200) is used in these examples.

7.1 – Case [-6, 3, 50]

The initial run of the decision tree is given the values of -6 for the deceleration threshold, 3 for the acceleration threshold, and 50 feet per second as the division of velocity between congested and free flow. Figure 18 is the resulting cluster plot using these settings. The number of patterns per clusters 1-4 is 16170, 9200, 14837, and 14241 respectively.

As seen by those numbers and by the plot, the congested region might be showing signs of start-stop waves as accelerating and decelerating patterns are partially visible amongst the congested patterns. This plot also has a slightly more distinguishable free flow region following the congestion (around 1800) than is seen in the previous space-time-cluster plots. This could be an indication that the decision tree method is assigning more values to free flow than *k*-means was assigning as shock and acceleration waves.

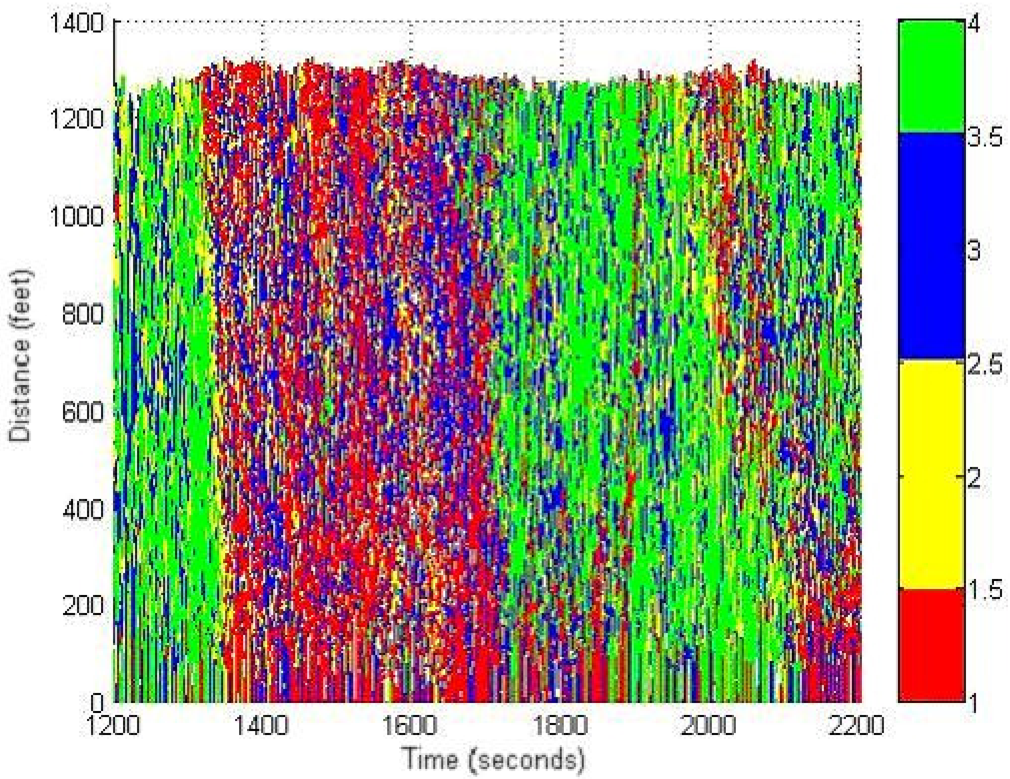

Figure 18 - Decision tree space-time-cluster plot for -6, 3, 50.

7.2 – Case [-6, 1.5, 50]

The first modified run puts an emphasis on acceleration, lowering the threshold value to see how many more are reclassified as being in the acceleration wave as opposed to the congested and free flow clusters. The changed leads to new cluster values of 12439, 9200, 21766, and 11043 for clusters 1-4 respectively. The acceleration wave cluster took roughly 3700 vehicles from the congested flow and roughly 2800 vehicles from the free flow.

The resulting plot, Figure 19, shows that the 6500 addition acceleration wave values tend to emphasize the attempts to reestablish free flow speed during the congested region. Furthermore, the reduced threshold value is forming a clearer forward shock as traffic reestablishes the free flow region. One attribute that is more visible is the presence of the forward shock seen in the KWM expectation graph, Figure 4, described Section 5.1. The free flow region reflects more acceleration wave vehicles than in Figure 18, but it is still giving a good look at the free flow region itself with fewer misclassified shock wave patterns present.

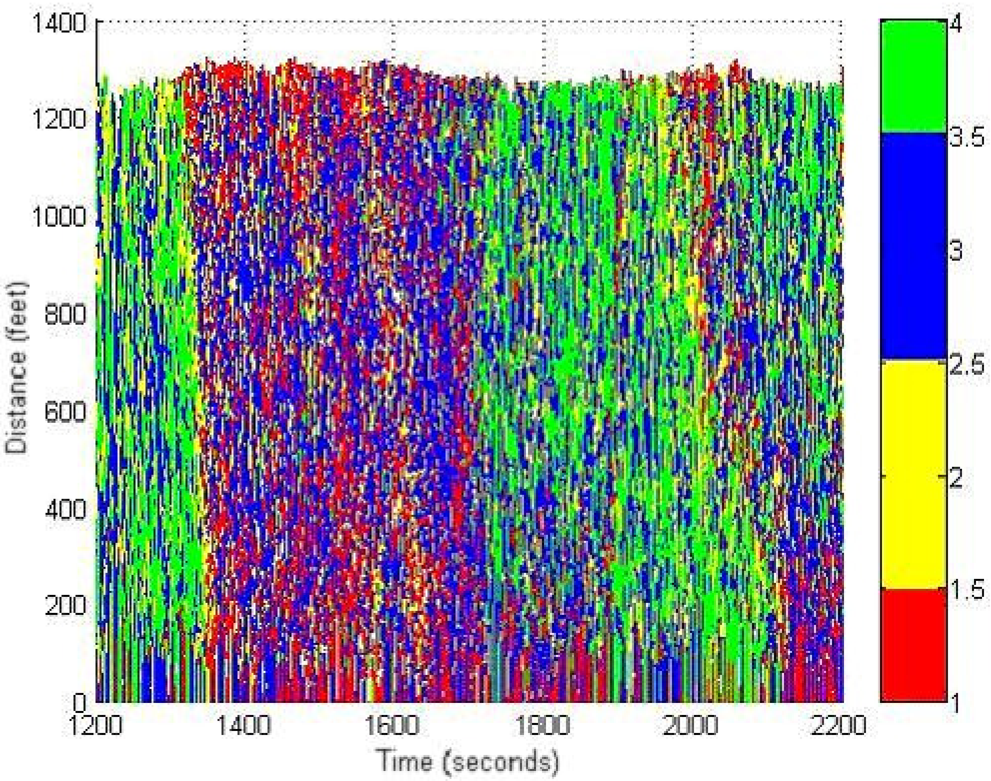

Figure 19 - Decision tree space-time-cluster plot for -6, 1.5, 50.

7.3 – Case [-6, 4.5, 50]

This execution of the decision tree takes the opposite approach, increasing the acceleration threshold and thus presumably reducing the number of values in the acceleration wave cluster and dispersing them into the congested and free flow clusters. The new number of patterns per cluster results in 17679, 9200, 12008, and 15561 for clusters 1-4 respectively. That is roughly 1500 into the congested flow and 1300 into the free flow.

The resulting Figure 20 shows an understandable increase in the congested region, although the increase in congested vehicles from this threshold change is not as noticeably different from the base case run as the reduction in congested vehicles seen in the plot of an acceleration threshold of 1.5. The increased threshold seems to create a worse view of the section, indicating that a larger acceleration threshold value is heading in the wrong direction.

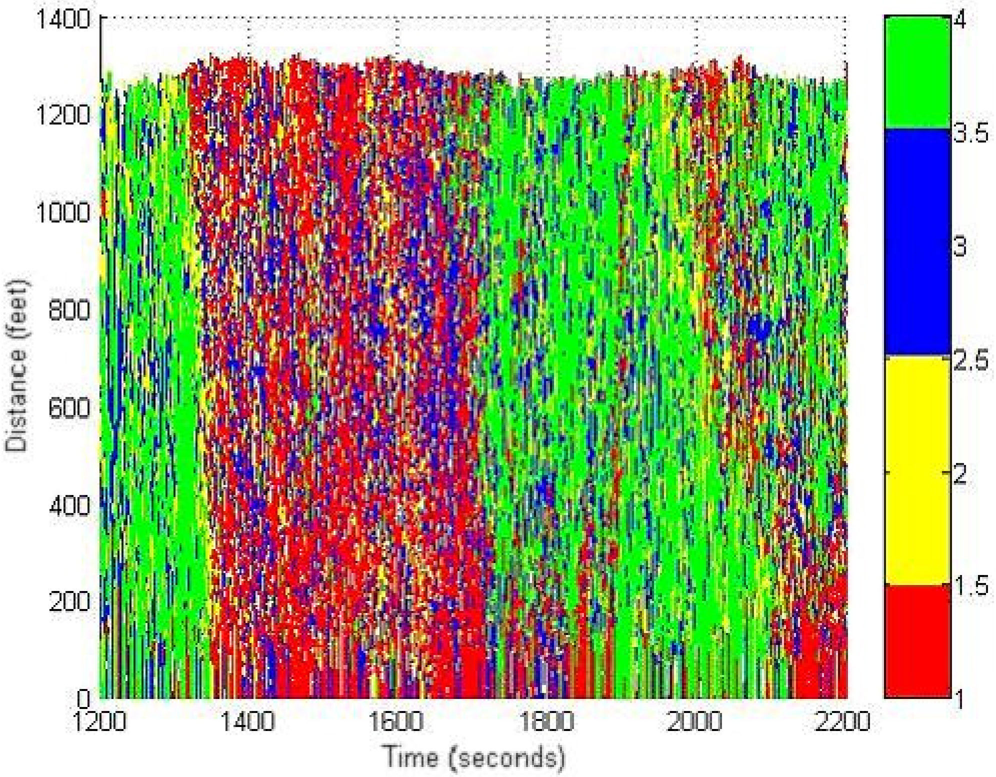

Figure 20 - Decision tree space-time-cluster plot for -6, 4.5, 50.

7.4 – Case[-10, 3, 50]

Starting with this section, the significant change is a reduction in the deceleration threshold value for all three of the acceleration experiments, as seen in Figure 21. In these cases, a driver's deceleration must be more significant to be considered a part of the deceleration wave (cluster 2). For clusters 1-4, there are 19374, 3375, 14837, and 16862 patterns respectively.

Nearly 6000 patterns are moved out of the deceleration wave cluster, with roughly 55% of them being inserted into the congested flow and the rest into the free flow clusters. It is within the congested region of the plot that the increase in the congested flow cluster size is most apparent; the amount of visible shock wave patterns is drowned out. With shock waves being harder to see, a larger negative threshold value for deceleration appears to be the wrong direction for adjustment.

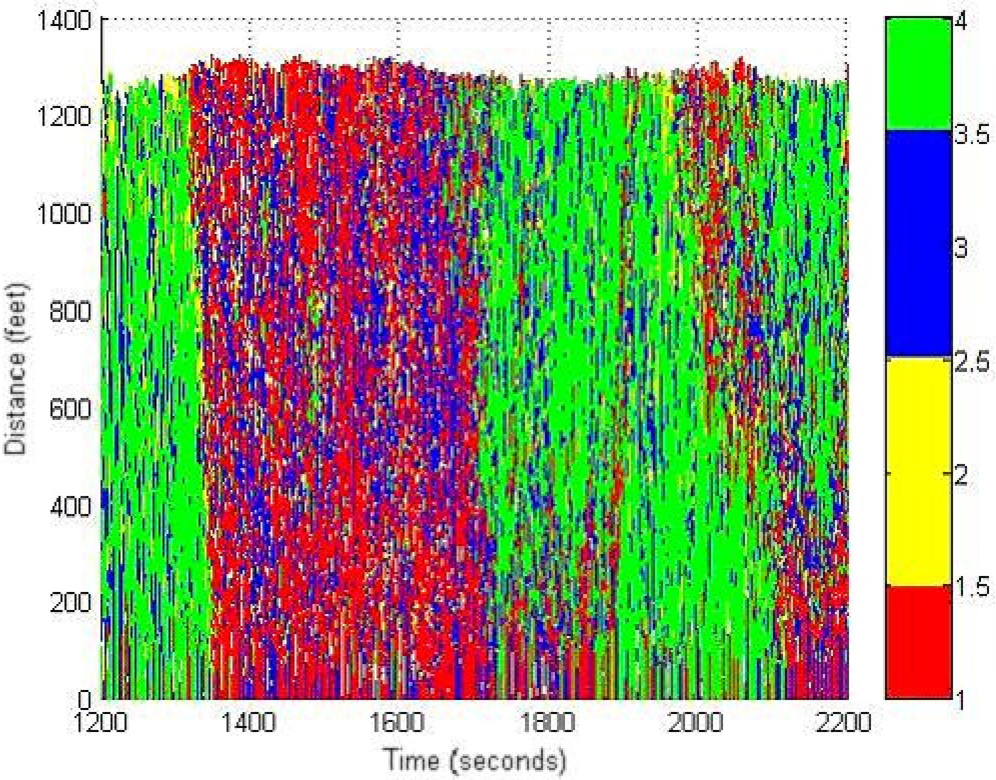

Figure 21 - Decision tree space-time-cluster plot for -10, 3, 50.

7.5 – Case [-3, 1.5, 50]

In both the deceleration and acceleration threshold cases, it appears that reducing the threshold may be the key to revealing more significant detail. With the new values, clusters 1-4 have a total of 7856, 17475, 21899, and 7218 patterns, respectively. The reduced number of patterns in both the congested and free flow clusters is apparent with both clusters containing less than 8000 patterns a piece versus the shock and acceleration wave clusters which now contain a much more noticeable number of patterns.

Figure 22 is the associated plot for this decision tree run. The initial shock wave preceding the congested region stands out much clearer in this case. The acceleration wave clusters perhaps are overly emphasized in this case, as there is a wide spread of blue amongst the congested and free flow regions in the time slice. The congested region itself is more difficult to define since the number of congested patterns is reduced by so much in this case.

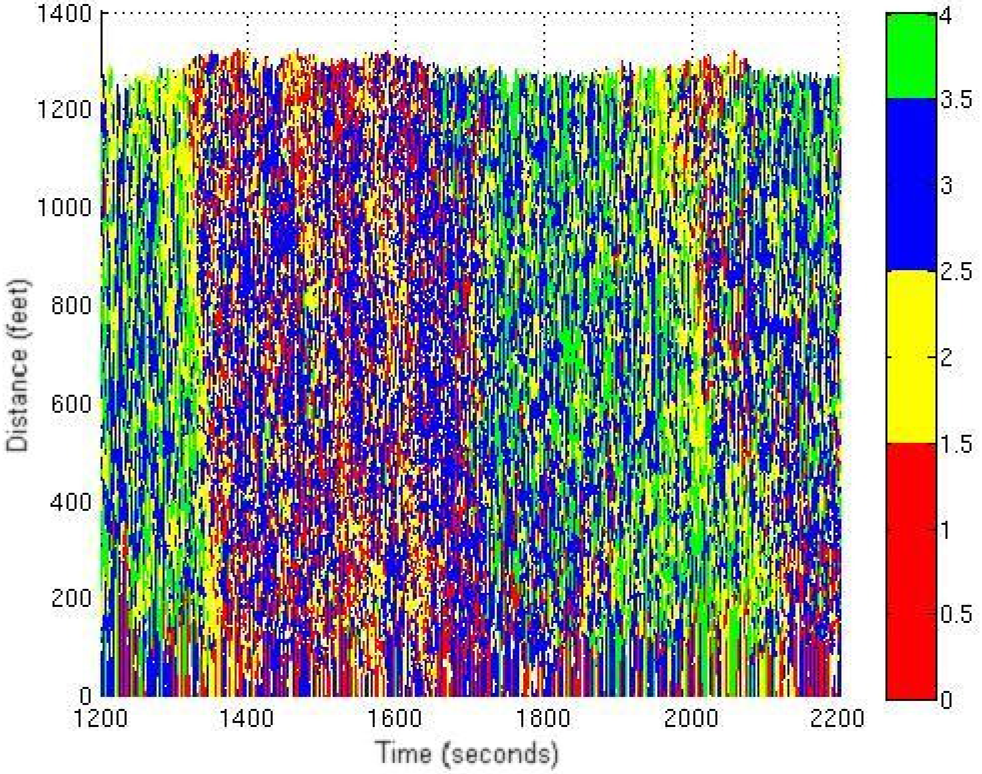

Figure 22 - Decision tree space-time-cluster plot for -3, 1.5, 50.

8. CONCLUSIONS AND FUTURE PROBLEMS

The included results show that the application of pattern recognition, specifically through clustering analysis in this case, provides a potentially useful method for examining and identifying the classes of traffic flow. This work structures data into a logical, efficient schema (in the form of a matrix) with the goal of assisting in efficient experimentation with microscopic traffic data, while avoiding specific data structures that would confuse or be incompatible with other styles already available. Instead of attempting to include all interesting features within the dataset itself this work focuses on storing and making available the key information needed to derive the more specific and interesting features one may wish to examine, and providing simple, object-like operators that have the ability to extract additional features that a particular researcher might wish to examine.

The *k*-means method, as implemented here with *k*=4 and in a highly supervised fashion, was not as useful as hoped in identifying the classes of traffic-flow patterns predicted by the classical kinematic-wave model, specifically in identifying spatio-temporal regions corresponding to shock and acceleration waves. As discussed in Section 6.1, this problem could be attributed to the use of training sets that select random patterns from the entire data set instead of patterns from time slices that more accurately reflect the experiment space (i.e., the time slice that our attempted data analysis focused upon). We nonetheless recommend further experimentation with this method, both with alternative values of *k*, and with implementation in unsupervised fashion.

The decision-tree method showed promise, for identification of the kinematic-wave classes, especially with lower acceleration thresholds that accent the acceleration waves exiting a congested region as well as a clearer free flow region with less patterns classified into the shock and acceleration waves (see Figure 19, and the associated discussion). The ability to adjust these thresholds is useful for quick examination of the quality of a particular plot without having to recreate each group of centroids with every adjustment, which is a downside in using the *k*-means approach. I recommend this approach be further developed.

Yet another approach to pattern recognition that might be useful for traffic flow is the method of expectation maximization [Dempster et al. 1977]. This approach seems well suited to data that do not cluster well in feature space, which is precisely the characteristic of traffic data, particularly under conditions of congestion (e.g., stop-and-go traffic). I accordingly recommend that its application to pattern recognition in microscopic data sets be further explored. (This possible approach was suggested to the author by Dr. Ricardo Guterriez-Osuna.)

The flexibility of the decision tree method in this work allows for a wide range of possible future applications. There are other features not discussed in this work that one could be interested in. Using the JHK datasets as an example, some of the interesting alternative features to examine are: the effect of lane changing effect on flow; vehicle following/leading distances effects on flow and acceleration/deceleration; and vehicle size (such as use in examining the effect of tractor trailers in a region of congested flow).

REFERENCES

CASSIDY, M. J. 1998. Bivariate relations in nearly stationary highway traffic. *Transp. Research B*, *B32,* 49-59.

DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, Series B, 39, 1-38.

DRAKE, J. S., SCHOFER, J. L., AND MAY, A. D. 1967. A statistical analysis of speed density hypotheses. *Highway Research Record*, 154, 53-87.

FU, K.S., MIN, P.J., AND LI, T.J. 1970. Feature selection in pattern recognition. *IEEE Trans. System Science and Cybernetics*, *SSC-6*, *1*, 33-39.

GREENSHIELDS, B.N. 1934. A study of traffic capacity. In *Proc. 14th Annual Meeting of the Highway Research Board*, 448-474.

HUANG, Y., MCCULLAGH, P.J., AND BLACK, N.D. 2004. Feature Selection via Supervised Model Construction. In *Proc. Fourth IEEE International Conf. on Data Mining (ICDM'04)*, 411-414.

KERNER, B.S. 2004. *The Physics of Traffic*. Springer, Berlin.

KERNER, B.S. AND KLENOV, S.L. 2002. Microscopic theory of spatial-temporal congested traffic patterns at highway bottlenecks. *Physical Review E* 68 036130-1 – 036130-20.

KOSHI, M., IWASAKI, M. AND I. OHKURA. 1981. Some findings and an overview on vehicular flow characteristics. *Proc. 8th International Symposium on Transportation and Traffic Theory (ISTTT)* (V. F. Hurdle *et al.*, Eds.), pp. 403-426, University of Toronto Press, Toronto.

LIGHTHILL, M. J. AND WHITHAM, G.B. 1955. On kinematic waves II – a theory of traffic flow on long crowded roads. In *Proc. Royal Society*, London, 317-345.

LIU, H., MOTODA, H., AND YU, L. 2004. A selective sampling approach to active feature selection. *Artificial Intelligence*, *159*, 49-74.

MACQUEEN, J. 1967. Some methods for classification and analysis of multivariate observations. *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1, 281-297.

NATIONAL CONSORTIUM ON REMOTE SENSING IN TRANSPORTATION (NCRST). 2007. http://www.ncgia.ucsb.edu/ncrst/ncgia.html

NEXT GENERATION SIMULATION (NGSIM): DATA SETS. 2007. http://www.ngsim.fhwa.dot.gov/modules.php?op=modload&name=News&file=article&sid=4

PENA, J. M., LOZANA, J. A., AND LARRANAGA, P. 1999. An empirical comparison of four initialization methods for the *K*-Means algorithm. *Pattern Recognition Letters, 20*, 1027-1040.

RICHARDS, P.I. 1956. Shockwaves on the highway. *Operations Research 4*, 42-51.

ROUSSEEUW, P.J. 1987. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Journal Computational and Applied Mathematics 20*, 53-65.

SCHÖCNHOF, M. AND HELBING, D. Empirical features of congested traffic states and their implications for traffic modeling. *Transportation Science,* submitted (2004). Preprint accessible at www.helbing.org, as of May 19, 2005.

SELIM, S.Z. AND ISMAIL, M.A. 1984. K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions of Pattern Analysis and Machine Intelligence 6*, 81-87.

SHAHAR, Y. AND MOLINA, M. 1998. Knowledge-based spatiotemporal linear abstraction. *Pattern Analysis and Applications 1*, 91—104.

SMITH, S.A. 1985. Freeway Data Collection for Studying Vehicle Interactions – Technical Report. Report No. FHWA/RD-85/108.

TRANSPORTATION RESEARCH BOARD. Highway Capacity Manual 2000, Washington, DC.

THE MATHWORKS, INC. 2007. http://www.mathworks.com/access/helpdesk/help/toolbox/stats/index.html?/access/helpdesk/help/toolbox/stats/silhouette.html&http://www.mathworks.com/cgi-bin/texis/webinator/search?pr=Support&db=MSS&prox=page&rorder=750&rprox=

750&rdfreq=500&rwfreq=500&rlead=250&sufs=0&order=r&support=Support&is_summary_on=1&ResultCount=10&query=silhouette&submit=Search

WEBB, A.R. 2002. *Statistical Pattern Recognition.* John Wiley & Sons, Ltd, West Sussex, England.

U.S. DOT. 1985. Freeway Data Collection for Studying Vehicle Interactions, Report No. FHWA/RD-85/108, JHK and Associates.

YOUNG, T.Y. AND CALVERT, T.W. 1974. *Classification, Estimation, and Pattern Recognition.* American Elsevier Publishing Company, Inc, New York.

APPENDIX A

NEXT.M

```
% next.m

% Returns the position in the matrix of the next (future) instance in
which
%   vehicle j shows up.

N=-1; % If NEXT is still -1 after search, searh did not find
appropriate value
j_p=j+1;

if(j==181148)
    j=181147;
end

time_j=traffic_data(1,j);
v_id_j=traffic_data(2,j);

while traffic_data(1,j_p)<=((time_j)+1)
    if traffic_data(2,j_p)==v_id_j
        N=j_p;
        break
    else
        j_p=j_p+1;
        if (j_p>181147)
            break
        end
    end
end

time_j=0;
v_id_j=0; % Zeroing out to clear variables in memory for next use.
```

PREVIOUS.M

```
% previous.m

% Returns the position in the matrix of the previous (past) instance in
%   which vehicle j shows up.

P=-1; % If PREVIOUS is still -1 after search, searh did not find
appropriate value
j_p=j-1;
```

```
time_j=traffic_data(1,j);
v_id_j=traffic_data(2,j);

% modification - re-examine after data_graphing experiment
if(j_p==0)
    j_p=1;
end

while traffic_data(1,j_p)>=((time_j)-1)
    if traffic_data(2,j_p)==v_id_j
        P=j_p;
        break
    else
        j_p=j_p-1;
        if (j_p<1)
            break
        end
    end
end

time_j=0;
v_id_j=0; % Zeroing out to clear variables in memory for next use.
```

LEADER.M

```
% leader.m

% Returns the position in the matrix of the vehicle that is leading
%   vehicle j in the same lane during a specific instance of time.

L=-1; % If LEADER is still -1 after search, searh did not find
appropriate value
j_p=j-1;

time_j=traffic_data(1,j);
lane_j=traffic_data(5,j);

while traffic_data(1,j_p)==time_j
    if traffic_data(5,j_p)==lane_j
        L=j_p;
        break
    else
        j_p=j_p-1;
        if (j_p<1)
            break
        end
    end
end

time_j=0;
lane_j=0; % Zeroing out to clear variables in memory for next use.
```

FOLLOWER.M

```
% follower.m

% Returns the position in the matrix of the vehicle that is following
%   vehicle j in the same lane during a specific instance of time.

F=-1; % If FOLLOWER is still -1 after search, searh did not find
appropriate value
j_p=j+1;

time_j=traffic_data(1,j);
lane_j=traffic_data(5,j);

while traffic_data(1,j_p)==time_j
    if traffic_data(5,j_p)==lane_j
        F=j_p;
        break
    else
        j_p=j_p+1;
        if (j_p>181147)
            break
        end
    end
end

time_j=0;
lane_j=0; % Zeroing out to clear variables in memory for next use.
```

ACCELERATION.M

```
% acceleration.m

% Returns the calculated acceleration for a vehicle using its previous,
% current, and next longitudinal positions.  Operators NEXT and
PREVIOUS
% are used to find the proper locations.

NEXT();
PREVIOUS();
if(N==-1|P==-1)
    A=-1000; % When acceleration cannot be calculated, set to -1000.
else
    A=(traffic_data(4,N))-(2*(traffic_data(4,j)))+(traffic_data(4,P));
% Delta_t omitted for being 1
end

N=0;
```

```
P=0; % Zeroing out to clear variables in memory for next use.
```

VELOCITY.M

```
% velocity.m

% Returns the calculated velocity for a vehicle using its previous and
next
% longitudinal positions to determine current velocity.  Operators NEXT
and
% PREVIOUS are used to find proper locations.

NEXT();
PREVIOUS();
if(N==-1|P==-1)
    V=-1; % When velocity cannot be calculated, set to -1 for now.
else
    V=((traffic_data(4,N))-(traffic_data(4,P)))/2;
end

N=0;
P=0; % Zeroing out to clear variables in memory for next use.
```

VITA

Name: Matthew James Fields

Address: Department of Computer Science, c/o Dr. Paul Nelson, TAMU MS
3112, College Station, TX, 77843

Email Address: mjfields@neo.tamu.edu

Education: B.S., Computer Science, Trinity University, 2003
M.S., Computer Science, Texas A&M University, 2007