



Delft University of Technology

A compact and scalable representation of network traffic dynamics using shapes and its applications

Krishnakumari, Panchamy; Cats, Oded; van Lint, Hans

DOI

[10.1016/j.trc.2020.102850](https://doi.org/10.1016/j.trc.2020.102850)

Publication date

2020

Document Version

Final published version

Published in

Transportation Research Part C: Emerging Technologies

Citation (APA)

Krishnakumari, P., Cats, O., & van Lint, H. (2020). A compact and scalable representation of network traffic dynamics using shapes and its applications. *Transportation Research Part C: Emerging Technologies*, 121, 1-19. Article 102850. <https://doi.org/10.1016/j.trc.2020.102850>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

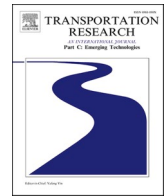
Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.



Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

A compact and scalable representation of network traffic dynamics using shapes and its applications

Panchamy Krishnakumari^{*}, Oded Cats, Hans van Lint

Department of Transport and Planning, Delft University of Technology, Stevinweg 1, 2628CN Delft, the Netherlands

ARTICLE INFO

Keywords:

Nation wide
Shapes
Network traffic dynamics
Pocket of congestion

ABSTRACT

The biggest challenge of analysing network traffic dynamics of large-scale networks is its complexity and pattern interpretability. In this work, we present a new computationally efficient method, inspired by human vision, to reduce the dimensions of a large-scale network and describe the traffic conditions with a compact, scalable and interpretable custom feature vector. This is done by extracting pockets of congestion that encompass connected 3D subnetworks as 3D shapes. We then parameterize these 3D shapes as 2D projections and construct parsimonious feature vectors from these projections. There are various applications of these feature vectors such as revealing the day-to-day regularity of the congestion patterns and building a classification model that allows us to predict travel time from any origin to any destination in the network. We demonstrate that our method achieves a 44% accuracy improvement when compared against the consensus method for travel prediction of an urban network of Amsterdam. Our method also outperforms historical average methods, especially for days with severe congestion. Furthermore, we demonstrate the scalability of the approach by applying the method on the entire Dutch highway network and show that the feature vector was able to encapsulate the network dynamics with a 93% prediction accuracy. There are many paths to further refine and improve the method. The compact form of the feature vector allows us to efficiently enrich it with more information such as context, weather and event without increasing the computational complexity.

1. Introduction

Understanding traffic dynamics has been one of the main research topics in the transportation sciences since the early 1950s. The focus of many of these studies is on describing and understanding the longitudinal dynamics of traffic along corridors (Gazis et al., 1959; Gipps, 1981; Prigogine and Herman, 1971; Helbing and Treiber, 1998; Nagel and Schreckenberg, 1992; Lighthill and Whitham, 1955; Payne, 1971; Whitham, 1975). However, the literature on modeling network-level traffic dynamics is limited, even though simulating traffic on this coarser level offers many opportunities for better understanding and managing traffic flows in large urban networks. The idea of a macroscopic fundamental diagram (MFD) (Godfrey, 1969; Mahmassani et al., 1987), along with the empirical evidence of its existence (Geroliminis and Daganzo, 2008; Leclercq et al., 2014; Yildirimoglu et al., 2015), provided a breakthrough in modeling such network dynamics (Haddad et al., 2013; Ge and Fukuda, 2019). It was found that details at the individual link level are not needed to describe the congestion dynamics of cities but can be instead defined based on homogeneous regions of a city. This was further extended in Lopez et al. (2017a), where these homogeneous regions are defined over both space and time simultaneously.

^{*} Corresponding author.

E-mail addresses: p.k.krishnakumari@tudelft.nl (P. Krishnakumari), o.cats@tudelft.nl (O. Cats), j.w.c.vanlint@tudelft.nl (H. van Lint).

In Lopez et al. (2017a), they generalized the 2D partitioning efforts using *snakes* (Saeedmanesh and Geroliminis, 2016) to 3D and proposed using more computationally efficient data point clustering methods such as k-means, Growing Neural Gas (GNG) and DBSCAN, followed by a post-treatment methodology to ensure that the clusters are connected in space and time. Moreover, these partitioning techniques were used to synthesize the network of Amsterdam over 35 days into 4 so-called consensual patterns with each 9 homogeneous 3D subnetworks and show that with these patterns the travel time of 84% of all trips between any OD in the network can be predicted with an error margin below 25% (Lopez et al., 2017b). Thus, Lopez et al. (2017b) demonstrates that investigating higher-level clustering over space and time can efficiently reduce dimensionality. However, the consensus partition is proved to be a NP-complete problem (Dörnfelder et al., 2014), which implies that the computational complexity, and thus time, of the algorithm increases rapidly as the size of the problem grows. Consequently, the consensus clustering becomes non-viable for large-scale urban networks.

To this end, we propose a new scalable approach, taking inspiration from human vision. The inspiration comes from the fact that humans use shapes, color and different physical characteristics for detecting faces and expressions. Instead of processing all possible shapes, we only need a general shape to detect a face. However, for face recognition, we might need more detailed physical attributes. Thus, depending on the application, we only need a general shape of the congestion propagation to distinguish between the different types of congestion. In Krishnakumari et al. (2017), the variation of two base shapes, identified based on well-established literature (Kerner et al., 2004), was proven to be sufficient for distinguishing between different types of congestion in corridor-level analysis with a 70% prediction accuracy rate. However, in network-level analysis, we do not have extensive knowledge of the different congestion patterns. Hence, in this work, we identify the recurrent shapes in network-wide congestion patterns and use these to distinguish between the traffic dynamics of a large-scale network. The main contribution of this work is a compact customized feature vector based on these identified shapes to define the daily traffic dynamics of a large-scale network. This feature vector can easily be extended to include more context information without further increasing the computational complexity.

The paper is organized as follows: Section 2 describes the overall methodology including the data preparation, feature vector formulation, and its applications. In Section 3, we outline how we assess the method. We demonstrate the method for two networks: a small network for which we compare the approach with three methods including the state-of-the-art consensus method and a larger network to demonstrate its scalability. We also briefly reiterate the consensus clustering approach proposed in Lopez et al. (2017b) in Section 3. The results are presented in Section 4. We then offer conclusion and a discussion on further research avenues in Section 5.

2. Method

The objective of this work is to build a compact, customizable feature vector - based on high-level features - that represent the traffic conditions of a large-scale network. To build such a feature vector, we first need to have a representation that can encompass the network traffic dynamics completely. We use the network and associated traffic variables to build 3D maps that incorporate the

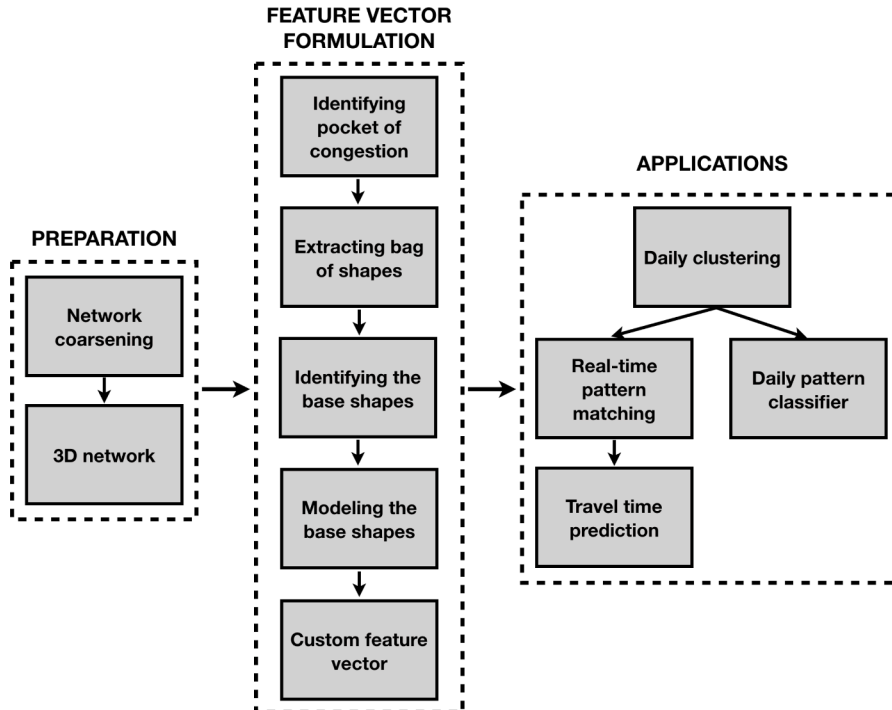


Fig. 1. Methodological framework.

spatiotemporal relationship of traffic as indicated in Fig. 1(a). There are different ways to derive feature vectors from network traffic dynamics white box methods (model-based), black-box methods (eg: neural networks), and hybrid methods. In this work, we are employing a hybrid method that builds a compact feature vector based on domain knowledge and derived from data. This will allow us to create interpretable feature vectors that tell us why a model fails and why it succeeds, unlike black-box methods.

We use high-level physical features such as shapes to construct these feature vector. For this, we first need to detect and extract congestion patterns from the 3D maps. We can extract shapes from these patterns and thus, each day can be represented by a collection or bag of shapes as there might be multiple patterns within a day. However, there are too many shapes within a given day and many more when we look at dynamics over multiple days. Hence, we use the assumption that we only need a general shape to describe the main process. For example, we only need the information of a general shape, color and other high-level physical attributes for detecting faces, whereas for face recognition, much more detailed features for distinguishing between two different people are needed. We extend this theory to traffic and hypothesize that, depending on the application, we only require high-level features such as the general shape of the congestion propagation to distinguish between different daily dynamics. Thus, each daily network traffic pattern can be represented based on these general shapes and their variations. These steps are used to construct the compact feature vector representation as shown in Fig. 1(b). Finally, we apply these feature vectors to reveal the regularity within the daily traffic patterns of an urban network and for travel time predictions as indicated in Fig. 1(c). The travel time prediction is used to validate that the method returns meaningful patterns that can provide accurate estimates.

2.1. Network and data preparation

We use the standard notation in graph theory to represent the transportation network. Here, the graph $G = (V, E)$ is a weighted directed graph where V is the set of nodes and E is the set of ordered pairs of edges or links. Each of the links is associated with a weight $w(u, v)$, where $(u, v) \in E$ and $u, v \in V$. In general, the weight can correspond to link length, width, flow or speed. However, since we aim to identify the pocket of congestion, hereby link weights correspond to speed. Note that our whole methodological framework uses a directed network representation. However, the opposing direction is not visible in the visualisations since the results are superimposed.

2.1.1. Network coarsening

As part of network preparation, we first reduce the complexity of the network as this is the first roadblock in a network-wide study. Network complexity can sometimes determine the viability of a method for a particular city. There are various coarsening schemes to reduce the complexity of graphs from the field of experimental algorithms (Brandt and Ron, 2013; Chevalier and Safo, 2009; Safo et al., 2015; Hamedmoghadam et al., 2019). A recent study focused on transport network offers a coarsening scheme more tailored for transportation network (Krishnakumari et al., 2019). The general idea is that, given graph $G(V, E)$ with n nodes, a more compact representation of the original graph, $G'(V', E')$, with a smaller number of nodes can be found which preserves its topological properties such as shortest path and network length. This is achieved by collapsing together nodes that have links attached to it with similar weights. A variance threshold is used to control the range of this similarity. A threshold of 0 implies that only links with the same

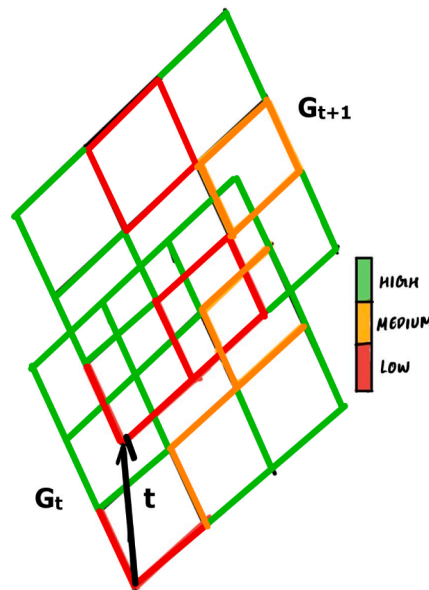


Fig. 2. 3D network with unidirectional virtual links connecting the 2D networks and representing time t where different colors display the edge weight - link speed - ranging from low to high speed.

weight are collapsed together. For more details on the coarsening algorithm, we refer the interested reader to (Krishnakumari et al., 2019). The authors also offer an open-source implementation of the algorithm (Opensource heuristic coarsening code). In this paper, we use the link speed as the weights for collapsing the node and a variance threshold of 0 is used to preserve the data for generating the coarsened graph.

Note that the coarsening is performed in order to reduce the computational cost. It is not an integral part of our proposed approach. The scalability of our approach refers to the ability of the method to compress the millions of data points into a low dimensional custom feature vector.

2.1.2. 3D network

Given the coarsened network $G'(V', E')$ and the associated link speed, we can incorporate another important characteristic of traffic - its temporal dynamics. Thus, instead of static graphs $G'(V', E')$, we need to construct time-dependent graphs $G'_t(V', E')$ to represent the network traffic dynamics in both space and time. The link speed is inherently time-dependent and can be represented as $w_t(u, v)$, where $u, v \in E'$. The resolution of the time slice t depends on the aggregation of the link speed. To construct this 3D map, directed weighted graphs for different time t are stacked together with unidirectional virtual links between the time slices, which only goes forward in time and not backward as shown in Fig. 2. The result is a very compact representation of the traffic dynamics in space and time.

2.2. Feature vector formulation

Customized high-level physical features coupled with traffic variables and other contextual information have proven to be successful in revealing regularity in corridor-based traffic congestion patterns (Krishnakumari et al., 2017). In this work, we extract the congestion shapes, identify the recurrent shapes and use this knowledge to build a feature vector that encompasses the network dynamics of an entire day of a large scale urban network. An overview of the steps for extracting these high-level features is given in Fig. 1 (b).

2.2.1. Identifying pockets of congestion using conditional 3D partitioning

In traffic networks, congestion propagates over space and time. Our main idea is to use these congestion regions to represent the dynamics of a specific day. For this, we first identify low speed regions from the weighted 3D network $G'_t(V', E')$. At any given time, most likely, a large portion of the network will not be congested (except in the case of severe gridlock). Therefore, to reduce the dimensionality of the problem, we do not consider the free-flow regions to define the network dynamics. To this end, we use a speed threshold, v^0 , to differentiate between congested and free-flow regions. In Krishnakumari et al. (2017), a threshold of 65 km/hr was used as the threshold for highways. So, any link with a speed lower than 65 km/hr will be considered as a congestion region. Based on the case study area, this threshold can be adjusted. This value, which heavily depends on the case study area, can be made relevant for any road type by using a relative link speed ratio instead of an absolute link speed. The speed ratio of a link at a given time t is the ratio between the speed drop (difference between the speed of the link at time t and its speed limit) and the speed limit of the link. Thus, a threshold of 0.5 on the speed ratio implies a speed drop of 50 % relative to its speed limit, which is significant and can be objectively categorized as a congestion region. However, this requires additional data regarding the speed limit of the links. Thus, depending on data availability, we can use a threshold v^0 on either speed or speed ratio to distinguish between congested and free-flow conditions as illustrated in Fig. 3(b).

Given that we have identified the free-flow and congested regions of the 3D network, we can now identify the pockets of congestion in the network. We define a *pocket of congestion* as a connected congestion region in the network, which is most probably, caused by a common bottleneck(s) or incident (Knoop et al., 2015). For this, we partition the heterogeneous congested regions in the network into homogeneous congestion zones. This clustering step is needed as it identifies different levels of congestion within a network, for example, light, medium or heavy congestion as illustrated in Fig. 3(c). Thus, each of the homogeneous regions can be characterized by

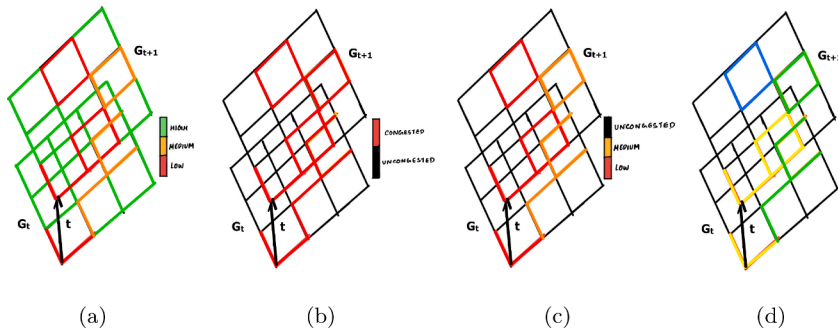


Fig. 3. Steps for identifying pockets of congestion from 3D speed maps (a) 3D network with colors representing link speed ranging from low to high speed (b) links categorised as congested and uncongested (free-flow) (c) congested links clustered into different unconnected zones based on their speed (d) extract the three connected zones which corresponds to the three pockets of congestion.

more or less equal speed. 3D network partitioning has been studied before in Lopez et al. (2017a,b), Gu and Saberi (2019), which proposes different 3D clustering techniques and post-treatment methodology for partitioning the entire network into homogeneous connected zones. However, they aim to partition the entire network to extract N connected zones from unconnected clusters, whereas we aim to find all the connected congested zones. Hence, we do not require any optimization in the post-treatment methodology to make sure congestion regions are connected. This improves the efficiency of the method exponentially. To this end, we extract the pocket of congestion by:

- Clustering the spatiotemporal link speeds in the congested regions.
- Connecting the resultant clusters in space and time.
- Each connected 3D cluster is represented by a single traffic variable, which can either be the mean speed or mean speed ratio of the 3D cluster depending on data availability.

The vectorised link speed $w_t(u,v)$, for all $u, v \in E'$ and for all t on a given day, is clustered into N unconnected clusters using k-means (MacQueen et al., 1967). The optimal number of N can be found by conducting a sensitivity analysis whereby we minimise the speed variance within the cluster while maximising the speed variance between the cluster. An in-depth sensitivity analysis of different clustering methods and for different N is provided in Lopez et al. (2017a). Since k-means does not incorporate spatial connectivity when clustering into different groups, we incorporate a simplified post-treatment methodology to extract zones that are connected in both space and time. This is done by finding connected components (CC) from the resultant k-mean clusters using depth-first search algorithm (Tarjan, 1972). A connected component is a sub-graph in which any two nodes are connected to each other by at least one path. Given the N clusters from the k-means clustering, there might be more than one CC for each cluster as shown in Fig. 3(c). All CCs within each cluster are identified and each of the identified CC is assigned a new cluster number. Thus, each of the identified CC is a pocket of congestion represented by a single variable - the average speed of the links that belongs to that cluster. The result is a very compact representation of the same spatio-temporal dynamics contained in the 3D network, but now in the form of 3D pockets of congestion represented by zone variables (average speeds). An example of the identified pocket of congestion is shown in Fig. 3(d).

2.2.2. Extracting bag of shapes

We extract 3D shapes of each pocket of congestion in a daily pattern by creating a convex hull around each of these congested zones. Each pocket of congestion in a daily pattern is represented by 4 variables (x, y, t, s) where x and y are the geographical coordinates of the locations inside the zone, t is the time extent of that zone and s is the cluster number of the zone. The convex hull of a zone represents the boundary of the zone as shown in Fig. 4(a). Because of the complexity of a 3D shape, we project the 3D shape on a 2D plane by creating a convex hull around (x, t) and (y, t) to create two 2D shapes for each zone. Thus, if there are p 3D pockets of congestion in a daily pattern, there will be $2p$ 2D shapes. By collecting the $2p$ shapes from all the daily patterns, we can create a comprehensive *bag of shapes* that represents the entire data space as shown in Fig. 4(b).

Some of the congestion pockets are relatively small, i.e. the congestion spans only a single link or over a single time period. We can remove such small congestion regions by setting a lower bound for the area of the shape using a^0 so that we only consider congestion of significant spatial and/or temporal persistence.

2.2.3. Identifying the base shapes

The bag of shapes can increase rapidly with the increasing availability of data from more days. In Krishnakumari et al. (2017), archetype shapes, also known as base shapes, have been identified for corridor-based traffic patterns based on literature to solve this problem. *Base shapes* are the generic shapes that recurrently prevail in the data, whose variations can be used to represent other shapes

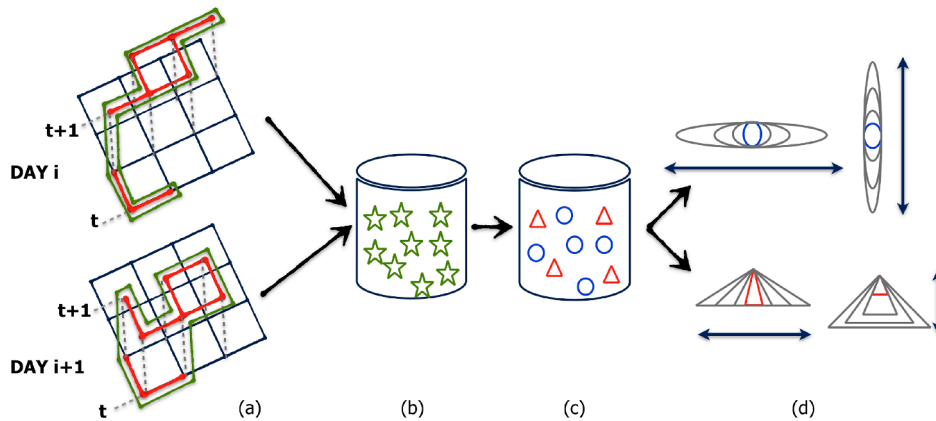


Fig. 4. Shape-based clustering (a) 3D convex hull from zones of all days (b) Bag of shapes (each star represents a 2D shape) (c) Clustering into base shapes (d) Base shape models where blue and red shapes correspond to mean shape and the gray corresponds to variations of the shape along their principle components. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in the data. For network-wide patterns, the shapes differ based on the network structure and underlying supply and demand conditions and hence, a straight forward manual base shape identification is not possible. Therefore, in this work, we introduce a data-driven approach to identify base shapes. This is done by defining a shape similarity index (SSI) for a pair of n -dimensional shapes x_1 and x_2 as:

$$SSI(x_1, x_2) = \sum_x \sum_y ||\hat{x}_1 - \hat{x}_2|| \quad (1)$$

where \hat{x}_1 and \hat{x}_2 are the superimposed shapes respectively, x and y the 2D coordinate space of the shapes. The shapes are superimposed on each other using Procrustes analysis (Kendall, 1989) which translates, rotates, and scales the shapes optimally to minimize the sum of squared distances, also known as Procrustes distance. Even though the shapes are derived from the same network, this step is necessary as similar shapes might be of different size and rotation. Registration algorithms like Procrustes correct for this by aligning these shapes so that they are comparable.

Based on the superimposed shapes, we next identify the base shape. The main purpose of identifying base shapes is to reduce the number of shapes used to represent congestion. Thus, given a congestion represented by (x, t) or (y, t) and if (x_1, t_1) from congestion 1 is the same as (y_2, t_2) from congestion 2, we only need one base shape to define both shapes. To identify these base shapes, the shape similarity index of all shape pairs in the bag of shapes is found. Thus, if there are $2p$ shapes in the bag of shapes for p 3D shapes, each shape is represented by $2p$ Procrustes distances and thus creating a $2p \times 2p$ distance matrix for all the shapes.

To estimate how many base shapes are required to approximate any shape in the bag of shapes, Principal Component Analysis (PCA) is applied to the distance matrix (Jolliffe, 2011). PCA gives the Eigenvectors and Eigenvalues of the distance matrix and we compute how many principal components are required to explain $v\%$ of the variances in the shapes. Thus, PCA allows us to identify the number of dominant shapes in the bag of shapes. In this work, we set v to 95%. The number of principal components, denoted by k , needed to explain 95% variance is used as the number of base shapes needed to represent the entire bag of shapes. Thus, the entire bag of shapes is clustered into k base shapes using k -means clustering as illustrated in Fig. 4(c). Consequently, we can create a separate bag of shapes for each base shapes based on the clustering resulting in k bag of shapes for k base shapes.

2.2.4. Modeling the base shapes

To model the variances within a base shape, an Active Shape Model (ASM) is constructed for each base shape. The input to the ASM algorithm is the bag of 2D shapes belonging to that base shape and the variance allowed. The ASM is introduced in Cootes et al. (1995) based on the principle that deformations of a shape can be represented by a so-called Statistical Shape Model (SSM) which contains all the parameters that are needed to define that shape. The SSM comprises of a mean shape and variations of that shape, analogously to a scalar value and statistical variations (variance) around that value. An example of mean shape and a couple of variations, also known as SSM components is illustrated in Fig. 4(d). Initially, a set of landmarks/points in the new shape is defined, after which the shape defined by these landmarks is deformed to provide the best fit available within the SSM. The deformation is based on finding correspondences between the new shape and the shape defined by different SSM components and iteratively minimizing a cost function for the fit. The allowed degree of deformation is constrained by the variations defined in the SSM. If the SSM includes large variances, large deformations are possible and vice versa. A more detailed explanation of ASM can be found in Cootes et al. (1995).

An overview of the ASM method for building a shape model for the base shapes are given below:

1. To build a shape model, all the shapes need to have the same number of dimensions. For this, we downsample all the shapes belonging to a given base shape to the minimum number of dimensions in the corresponding bag.
2. The downsampled shapes are aligned to the first shape in the bag as an initialization step using Procrustes analysis. An initial mean shape is generated from the aligned shapes.
3. The downsampled shapes are realigned to the mean shape and a new mean shape is generated from the aligned shapes. Repeat this step until convergence to generate the SSM mean shape.
4. Apply PCA on the final aligned shapes to compute the Eigenvectors and Eigenvalues which constitutes the SSM components and their variances respectively. Thus, any shape x can be approximated using the SSM mean shape \bar{x} , SSM principal components P and the variances b that deforms the shapes according to the components as

$$x \approx \bar{x} + Pb \quad (2)$$

5. Repeat the steps for all base shapes.

Thus, we identify the base shapes from the bag of shapes and build an SSM model for each of them.

2.2.5. Custom feature vector

We now have the necessary ingredients to define a compact yet insightful feature vector to represent the network traffic dynamics of a daily pattern. Given the $2p$ 2D shapes of the daily patterns, we can use the shape model to categorize these shapes into the base shapes to create a multidimensional feature vector $(n_1, n_{i+1}, n_{i+2}, \dots, n_{i+m})$; where n_i represents the number of occurrences of base shape i in that daily pattern and m is the total number of base shapes identified. The 2D shapes are categorized into i^{th} base shape by minimizing the ASM fitting error between the shape models of all the base shapes. The ASM fitting process is as follows:

1. Given a new shape Y' , we downsample the shape to the same dimension as the mean shape \bar{x}
2. Assign shape parameter b as 0.
3. Initialise the model point x based on b using $x = \bar{x} + Pb$
4. Align the new shape Y' to x using Procrustes analysis.
5. Update b in (2) to match the new shape Y' by solving $Pb = Y' - \bar{x}$
6. Compute the error metric by finding the difference between the mean shape \bar{x} and the ASM fitted shape Y'
7. Repeat steps 3–6 until convergence. The error metric is used as the convergence criteria.

The new shape is fitted to all the base shapes and the corresponding fitting error of each of the base shape is computed. The base shape that minimizes the fitting error is chosen as the base shape that the new shape belongs to. Additionally, we also use the traffic variables within the shapes (average speed) to define the traffic state of that network. Instead of using the average speed, we define k clusters which represent k ranges of speed in order to be consistent between different days and also to interpret the feature vector easily. To be consistent, we assign k as the number of clusters N used for clustering the congested regions into different speed regions in Section 2.2.1. The final feature vector that fully defines the dynamics of a day based on the congested regions is defined as follows:

$$FV_{day} = [n_{i+1}, \dots, n_{i+k}, \dots, n_{i+m+1}, \dots, n_{i+m+k}, s_1, \dots, s_k, v_1, \dots, v_k] \quad (3)$$

where k is the number of speed clusters, m is the number of base shapes, n_{i+1} is the number of occurrences of base shape i in cluster 1 in a 3D pattern of day , s_1 is the average speed within speed cluster 1 and v_1 is the variance of speed within cluster 1. This feature vector is computed for all days.

2.3. Applications

There are different applications for representing traffic data using these compact feature vectors. We reveal the regularity between different days and use these insights to extract the essence of these days into a set of representative 3D speed maps. Here, we present two such applications. The first application is using these 3D maps for short-term predictions of congestion propagation, which can be used for the travel time predictions. The travel time prediction is used to evaluate the performance of our method against a comparable method. The second application is to reveal the representativeness of these feature vectors in revealing the day-to-day regularity by investigating the classification accuracy of the identified daily clusters. This application is also used to evaluate the scalability of our approach. In the following we first describe the daily clustering procedure before elaborating on the two aforementioned applications.

2.3.1. Daily clustering

K-means is used to cluster the days, represented using the custom feature vector, into M classes to reveal regularity between days. There are different approaches to find a representative 3D network for each class. We used the centroid of the 3D speeds maps of the days that belong to each of the classes as follows:

$$\lambda_m = \frac{1}{n} \sum_{i=1}^n \lambda_i \quad (4)$$

where λ_i is the single ordered vector of traffic observations in the 3D network whose values can either be the link speed or speed ratio, n is the number of days in class m . Thus, λ_m is calculated for all classes and we obtain a 3D map for all classes.

2.3.2. Real-time pattern matching and travel time prediction

Given the current speed observation, we match the observed speed observations to these representative 3D maps to predict to which class is the test day resembles most. This is done by matching the current traffic state to the 3D representative models to predict the next traffic state. For the matching, we need to find the class that best fits the new data. For this, depending on the size of the available real-time speed data, the model is truncated and a similarity matrix is computed between the real-time speed and truncated model of each class. The model that maximizes the similarity is chosen as the predicted speed map of the new data. Since the model of each class λ_m is mean speed values, we use the Euclidean distance between the truncated model and the current speed for computing the similarity matrix. The complete 3D model of the matched class is the predicted traffic state of the test data which can be used for travel time predictions. We evaluate the quality of the clustering and the matching process using this prediction error. For the evaluation, we conduct leave-one-out validation. For a given day, the following steps are performed without considering that test day:

- cluster the feature vector of all days other than the test day into M classes
- create the mean model of these classes based on the shape-based feature vector
- fit the speed profile of the day under consideration to the mean model to predict the future speed profile
- the predicted speed profile model is used to generate travel time for predefined routes

These steps are repeated for each day in the dataset. The prediction accuracy is evaluated by computing the travel time of the predefined routes through the fitted speed profiles and comparing the results to benchmark methods and ground truth travel time. The travel time of the predefined routes is computed every t minutes in the 3D map so as to get a more representative sample. An exhaustive

analysis of the travel time error is done to evaluate the prediction using basic performance indicators such as Mean Absolute Percentage Error (MAPE) and the Root Mean Squared Error (RMSE).

2.3.3. Daily pattern classifier

The aim of this application is to show that the clusters of daily patterns extracted based on the custom feature vector encompasses the entire data set. This is done by classifying a new network traffic pattern into one of the M predefined daily pattern clusters and investigate the classification accuracy. The feature vectors of the entire data set is clustered into M classes and this class labels is used as the ground truth. For estimating the classification accuracy, we used a leave-one-out strategy and the following steps are performed for a given test day:

- Cluster the feature vector of the days other than the test day into M classes using k-means.
- Train a simple k-means classifier with input as the feature vectors and the output as the estimated class number from the previous step.
- Estimate the class label of the test day by classifying the feature vector of the test day into M classes using the trained classifier.

The classification accuracy is measured as:

$$accuracy = \frac{\text{number of correctly labelled days}}{\text{total number of days}} * 100\% \quad (5)$$

We also use a confusion matrix (Fawcett, 2006) to investigate the performance of each class in order to get insights into these classes.

3. Experimental setup

3.1. Data

We demonstrate our methodology on two network. First, a relatively small network of the major streets within the city of Amsterdam (excluding the freeways), which consists of 7512 links and 6528 nodes. The network data are obtained from OpenStreetMaps. Second, our method is applied for the entire Dutch highway network with 131 994 links and 116 305 nodes, which is openly accessible from the Dutch road authority through Nwb (Nwb (nationaal wegenbestand)). These case studies show the viability of our approach for both small-scale and large-scale networks as well as for both urban and highway road network.

The traffic information available for the city of Amsterdam is comprised of mean speeds available every 10 min between 7 am and 3 pm (≈ 8 h) for all links during 35 days from 23 February 2015 to 4 April 2015 (except Sundays). This information is derived from license plate recognition system at different critical points of the network. The methodology to derive link speed data from trajectory data; creating the mapped network; and reconstruct missing data can be found in Lopez et al. (2017a). For the entire Dutch highway network, there are more than 10 000 detectors across the network which are placed approximately 500 meters apart. Most of these detectors collect both speeds and flow every 1 min. We use the well-known adaptive smoothing method introduced by Treiber and Helbing (Treiber and Helbing, 2002) to fill in the missing speed information between detectors which results in filtered data every 30s. Since the speed limit of the highway network is also available, we use the speed ratio instead of absolute speed to represent the dynamics of the highway network. We collected these data for 45 days from 1 June 2018, to 15 July 2018, between 03:00 and 23:59 (≈ 21 h).

For both case studies, we use coarsening to reduce network complexity. The urban network of Amsterdam is coarsened using the link speed at 16:00 (peak period) as the edge weights. The coarsened network contains 411 links. However, some of these links have missing data for some days. By taking the intersection of all links that have data for the 35 days, the final coarsened network contains 208 links and 214 nodes (a complexity reduction of 97%) forms a single component as shown in Fig. 5(a). Each day is represented by a 3D network of dimension 208×48 (number of links \times time periods). For the entire Dutch highway network, we use the entire speed ratio vector of the link as the link weight for coarsening. The final coarsened network contains 38 662 links and 34 655 nodes with 14 connected components as shown in Fig. 5(b), which is a $\approx 71\%$ reduction in the number of links with a 0 variance threshold. Thus, each day is represented by a 3D network of dimension 38662×2519 , which implies ≈ 97 million data points to represent a single day.

3.2. Evaluation

To evaluate our method, we use three methods including the current state-of-the-art in network-wide analysis - consensus method, which was found to be able to synthesize 35 days of data into 4 consensual patterns (Lopez et al., 2017b). The first benchmark method is a historical average model where the feature vector of a day is defined as:

$$FV_{day} = [s_{11}, \dots, s_{xt}] \quad (6)$$

where s_{11} is the speed of link 1 at time 1 and s_{xt} is the speed of link x at time t . Thus, the feature vector is an ordered vector of speed values. A class is defined for each day of the week, thus 7 classes if we have data from Monday to Sunday. The representative 3D network for each class λ_m is found by taking the average of the FV_{day} belonging to class m .

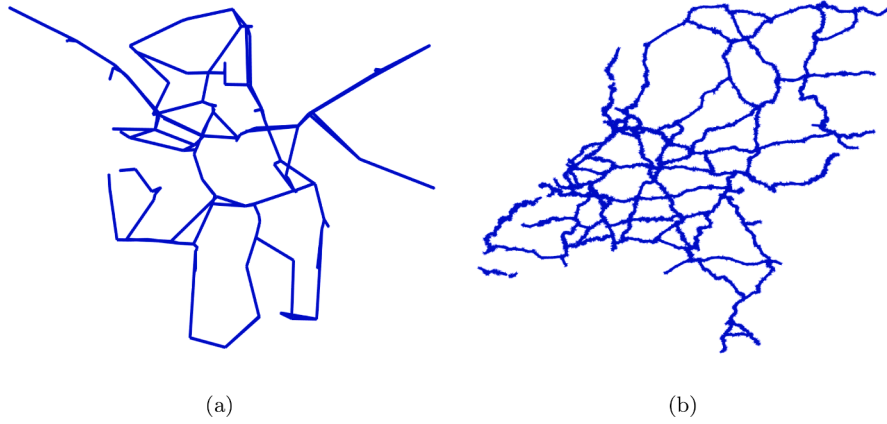


Fig. 5. Coarsened network of (a) Amsterdam with a single connected component (b) Entire dutch highway network with 14 connected components.

In the second benchmark model, we use the same feature vector as defined in (6), but instead of assigning the day of the week as the class, we use the k-means to cluster the days into M classes. Then, the representative 3D network λ_m is constructed using averaging as defined in (4). The pattern matching for the benchmark method follows the same step as defined in Section 2.3.2. Thus, both the historical average and data-driven historical average benchmark models use the full $x \times t$ dimensional speed values to construct the model compared to the low dimensional custom feature vector of the shape-based model.

The third method is the consensus method, which we briefly explain here, but for further details refer to (Lopez et al., 2017b). For extracting regularity between the daily patterns using the consensus method, there are several pre-processing steps:

- Constructing 3D network describing link-based traffic states
- Constructing 3D patterns describing zone-based traffic states
- Clustering 3D daily patterns using a consensus approach
- Real-time pattern matching for short-term travel time predictions.

The first step is the same as our data preparation module. The second step includes k-means clustering followed by a full-fledged post-treatment methodology which involves dividing the entire 3D network into k zones, which has a complexity of $O(N!)$ where N is the number of connected components in the 3D network after the initial clustering whereas our identification of pocket of congestion only has a complexity of $O(N)$.

The third step involves clustering the daily 3D patterns representing the zones. In Lopez et al. (2017b), a similarity measure known as normalized mutual information (NMI) has been used for the clustering. NMI has been extensively used for assessing the similarity between two clustering results (Cover and Thomas, 2012). NMI for comparing two 3D zones π_i and π_j of two different days is denoted by $NMI(\pi_i, \pi_j)$. π_i is a single ordered vector of all observations in the 3D zones whose values are the cluster ID. The pairwise NMI for all days is calculated and the Ncut algorithm (Shi and Malik, 2000) is used on this similarity measure to cluster the days into different classes.

For each of the classes, a representative 3D partition is obtained using consensus clustering. The two steps in consensus clustering determine the best partition from a set of partitions and improves this partition to obtain the consensus partition. In Lopez et al. (2017b), the following steps are performed to obtain a representative 3D median partition (consensus partition) for each class:

- A representative partition m is found for each class from the group of partitions in that class using BOK algorithm (Filkov and Skiena, 2004). This algorithm maximizes the total similarity TS with all the other days belonging to the same cluster.

$$TS = \sum_{k=1}^a NMI(\pi_m, \pi_k) \quad (7)$$

where a is the number of days in a given cluster, π_m and π_k is the representative partition and partition of every other day in the same cluster, respectively.

- The partition m is refined by randomly changing the label of the partition m and investigating whether that improves the TS. This is done using one element move (OEM) algorithm (Filkov and Skiena, 2004).

The details on clustering the individual days and creating the consensus partition are given in Lopez et al. (2017b).

The final step is the real-time pattern matching where the consensus model is matched with the current speed observation. In order to match the truncated consensus partition of each class to the new speed values, the following steps are applied:

- compute the speed vector to define the consensus partition of a class as follows:

$$\hat{s}(m, z) = \frac{1}{M*N*T} \sum_{i=1}^M \sum_{j=1}^N \sum_{k=1}^T s(i, j, k) \quad (8)$$

where $\hat{s}(m, z)$ is the mean speed of zone z for the consensus partition of class m , M is the number of historical days in class m , N is the links that belongs to that zone, T is the truncated time period and $s(i, j, k)$ is the speed of link j at time period k for the i^{th} day in class m .

- compute a new speed vector of the current traffic state as follows:

$$\bar{s}(m, z) = \frac{1}{N*T} \sum_{i=1}^N \sum_{j=1}^T s_{test}(i, j) \quad (9)$$

where $\bar{s}(m, z)$ is the mean speed of zone z for the consensus partition of class m for the new test day and $s_{test}(i, j)$ is the speed of link i at time j .

- compute the euclidean distance between $\hat{s}(m, z)$ and $\bar{s}(m, z)$ for all zones and all classes.

Thus, the class that fits the new data is the m that minimizes the Euclidean distance for all zones z .

We conduct a comparison study only for the urban network of Amsterdam as processing the data for the entire Dutch highway network using consensus partition will lead to an unreasonable amount of running time (\approx months). This is due to two reasons - the post-treatment methodology has a complexity of $O(N!)$ and the consensus partitioning was proved to be a NP-complete problem. Consequently, we demonstrate the first application on the Amsterdam network and the second application on the entire dutch network.

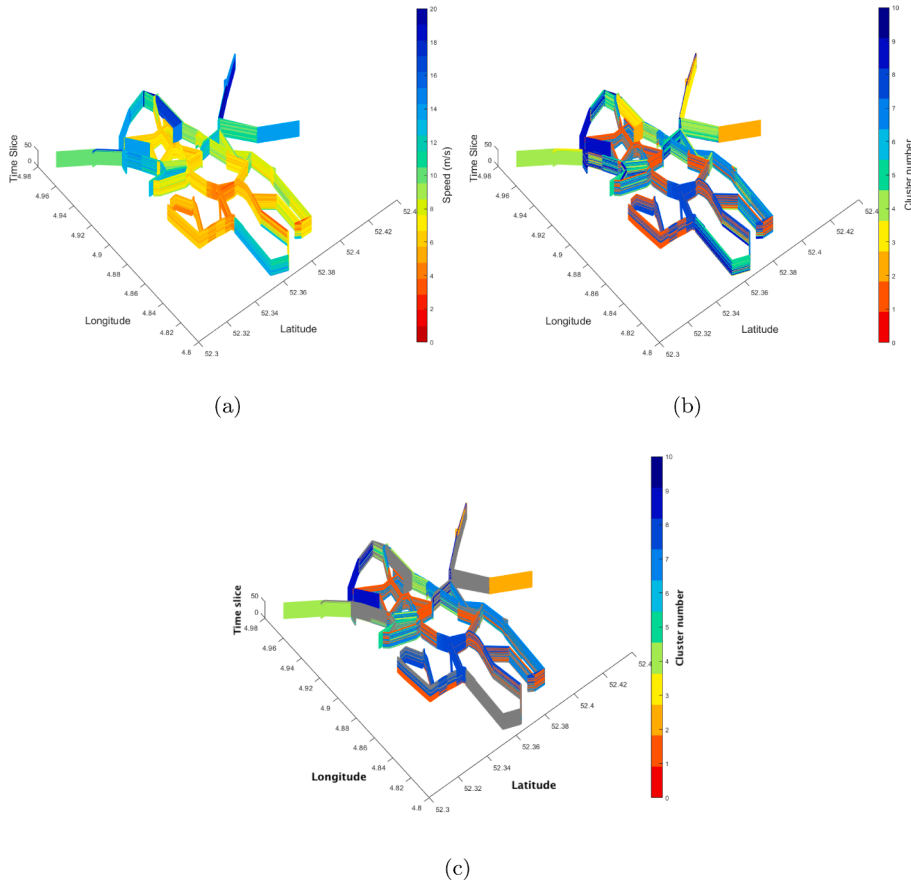


Fig. 6. Identifying pockets of congestion (a) 3D speed map (red implies low speed, blue high speed) (b) k-means clustering to extract unconnected clusters (c) Extract connected clusters to identify the top 10 pocket of congestion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Results

In this section, we present the results of the shape-based clustering for the two case study networks. We demonstrate the performance of the method by comparing the travel time prediction with the three methods. Then, we apply the method on the entire Dutch highway to reveal the regularity of network patterns.

4.1. Application 1 - Comparison with three benchmark models

The Amsterdam network is used for comparing our approach to three methods - historical average (benchmark 1), data-driven historical average (benchmark 2) and consensus method. An example of a 3D speed map of Amsterdam is shown in Fig. 6(a). To allow for a meaningful comparison of the consensus and shape-based approaches, we used the same number of zones to partition the network, which is set to 10. We use 10 3D zones to describe an individual day of 8 h for the consensus method. Since we do not have information about the road type and speed limit, we cannot set an informative v^0 to distinguish between congested and uncongested regions. Thus, we consider all the links as congested, which is the worst-case scenario in terms of computational time for the proposed method. The congested links are partitioned using k-means clustering with $k = 10$ as shown in Fig. 6(b). The pockets of congestion are identified from these unconnected clusters. The top 10 connected pockets of congestion for a given day are shown in Fig. 6(c). The top 10 refers to the largest connected components in terms of size with the lowest speeds. Interactive visualization can aid in making it easier to understand and interpret these 3D graphs.¹

The 3D shapes are extracted from the pocket of congestion as shown in Fig. 7(b). The projected 2D shapes with latitude \times time and longitude \times time are shown in Fig. 7(c) and (d), respectively. From the figure, we can see that there is regularity within the 2D shapes. Put simply, many shapes have similar geometries. There are a lot of recurrent shapes that can easily be described by a shape model rather than having separate shape models.

The 2D shapes from the 35 days are collected together to form the bag of shapes for Amsterdam. It was found that 95% of the shape variations in the bag of shapes can be represented by 2 base shapes and its shape models. Based on the similarity distance matrix, we clustered the bag of shapes into the two base shapes leading to the first base shape with 258 shapes and second base shape with 606 shapes. The shape model of each base shape contains a mean shape and the corresponding SSM components that can explain 95% of the variance within the shape. Base shapes 1 and 2 have 10 and 11 principal components, respectively. The corresponding mean shape and one of the SSM components of each base shape are shown in Fig. 8.

The shape models are used to construct the feature vector which along with additional traffic variable information is used for clustering the daily patterns into different groups. Since k is 10 and the number of base shapes is 2 for Amsterdam use case, the feature vector dimension is 40. Thus, a single day can be represented using 40 values instead of 9984 (number of links \times time resolution), leading to a dimensionality reduction of more than 99%. These feature vectors can be clustered to reveal regularity between days. In Lopez et al. (2017b), 4 consensus models are used to represent the 35 days. In this paper, we, therefore, use the same number of clusters to conduct a fair comparison for travel time prediction. Thus, the feature vectors are clustered into 4 classes and the mean speed model of each class using the shape-based approach is shown in Fig. 9.

The historical average benchmark method has 6 classes representing Monday to Saturday benchmark as Sunday was not included in the dataset. The number of classes for the data-driven average method is set to 4 as for both the shape-based method and the consensus method, the number of classes is 4. The results of the travel time prediction based for the benchmark, consensus, and shape-based models are presented here for the Amsterdam case. We conduct a leave-one-out strategy for the travel time prediction so that 34 days are used for training the classifier of the daily patterns and the remaining day is used for testing. This is repeated for all 35 days. For validating the travel time accuracy, 10 routes are randomly generated from the network. The ground truth is the travel time computed from the observed link speed from the 3D speed maps and we compare this against our results and against the travel time computed using the consensus models and the benchmark methods. Fig. 10 shows the 10 routes and an example result for a given day with the travel time estimated using the observed ground-truth speed, our shape method, the consensus method, and the two benchmark methods.

It can be seen that half of the travel times are underestimated for the routes using the shape-based method, albeit with a marginal error. In contrast, the other methods significantly overestimate the travel time, especially the benchmark methods. Moreover, both the historical average benchmark method and the data-driven benchmark seem to have a similar performance. This is also evident from Fig. 11(a-d), which shows a more detailed error distribution of the 10 routes for all 35 days. 64% of the travel time error for the shape-based method is because of an underestimated travel time (positive error time) compared to the negative error time performance of the other methods. Thus, the shape-based method seems to be able to encompass congestion dynamics slightly better than the prediction it yields under free-flow conditions. We consider its performance to be acceptable since it is particularly important to obtain good predictions in case of an incident or traffic jam.

Fig. 12(a) and (b) show the MAPE and RMSE results of the 10 routes for the 35 days, respectively. The shape-based method has a mean prediction error of approximately 8% compared to 14% for consensus, which is an improvement of around 44%. 95% of the travel time prediction has less than 9% error for the shape-based approach compared to 19% when applying the consensus method. Some of the days perform worse than others. There are three well-defined peaks in Fig. 12(a) and (b) for the shape-based and consensus

¹ <https://github.com/Panchamy/3D-Partitioning/wiki>.

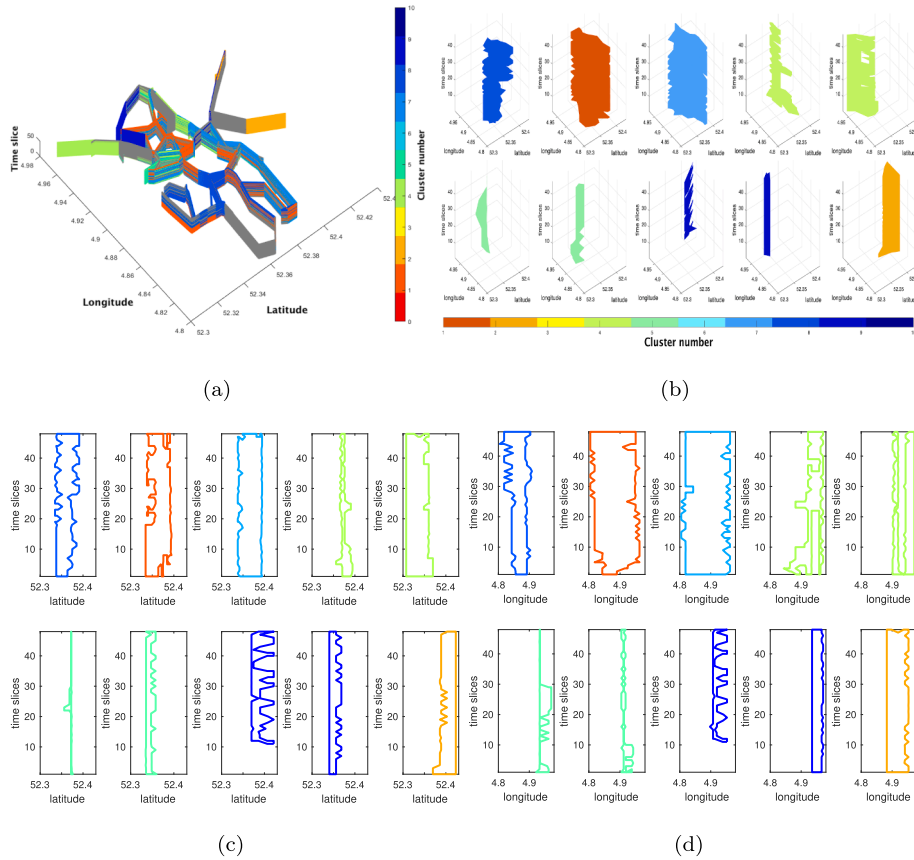


Fig. 7. Representing the top pockets of congestion for a given day as 2D shapes for the Amsterdam case study (a) top 10 pockets of congestion for that day (b) corresponding 10 3D shapes of the pockets of congestion (c) projected 2D shape representing the boundary between latitude and time (d) projected 2D shape representing the boundary between longitude and time.

methods. This is probably because of non-recurrent incidents or special events that occurred on these days. Particularly, the higher RMSE values are because of the difference in magnitude of the travel time for the congestion day vs a normal day. For example, on a normal day route 7 has a travel time of 13 min, whereas the congested day (test day 18) has a travel time of up to 100 min for the same route. A well-defined incident database can provide more insight into these irregularities in traffic dynamics, which can ultimately be used for improving travel time predictions under such conditions.

One of the limitations of our method is that the location of the congestion shape is not explicitly embedded within the custom feature vector. In order to examine the implications thereof, we have done a small qualitative analysis to check the relevance of the congestion location or lack thereof within the feature vector. For this, we have selected two arbitrary days belonging to the same class and compared the distribution of 3D shapes within these days as shown in Fig. 13. The hypothesis is that if the days belong to the same class, they should have a similar spatial-temporal distribution of the congestion patterns. From Fig. 13, it seems that even though our feature vector doesn't include location information, the method consistently finds the same large-scale patterns at the same locations (eg. shapes 1, 2, 3 in Fig. 13(a) is similar and at the same location as shapes 1, 2, 4 in Fig. 13(b)). However, within the small-scale patterns of the same day, we do observe some differences. If we add location information to the feature vector, we would presumably be able to also cluster similar days based on the similarity within small-scale patterns. However, a systematic analysis is needed to quantitatively verify this hypothesis by adding location to the feature vector and comparing the performance against the feature vector which does not embed the location.

As for the computational time for the shape-based and consensus methods, the steps that differ between the two methods are the post-treatment and building the models themselves. The post-treatment for all 35 days took approximately 1 s for the shape-based and 8 s for the consensus method. The computation time for building the consensus model is approximately 6 min whereas the shape-based method is less than 4 min. The computational improvement of the shape-based method seems insignificant for the Amsterdam use case. However, the computation times for both modules - building and post-treatment - increase rapidly with an increase in network size for the consensus method as we will illustrate in the next section.

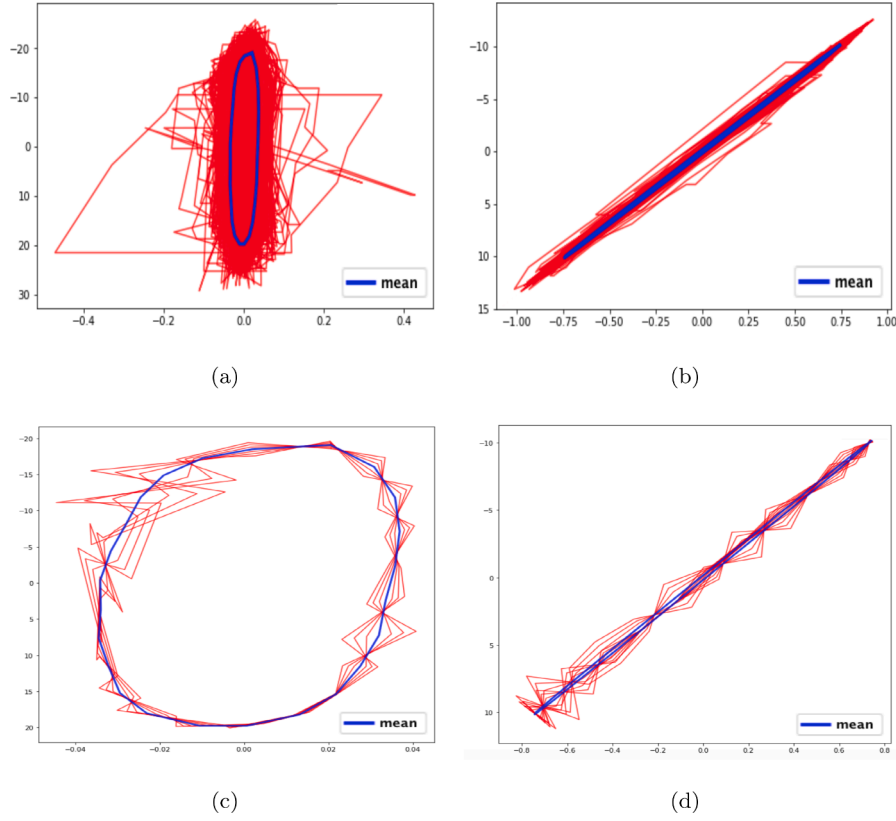


Fig. 8. Two base shapes of the Amsterdam case study derived from the projected 2D shapes with the mean shape highlighted in blue. The red shapes are the aligned bag of shapes in each base shape (a) and (b). (c) and (d) show one of the principal components of base shape 1 and 2, respectively. Note that the axes are dimensionless as they are created as a linear combination of latitude/longitude and time (which have two different units) and are standardized as part of the PCA procedure.

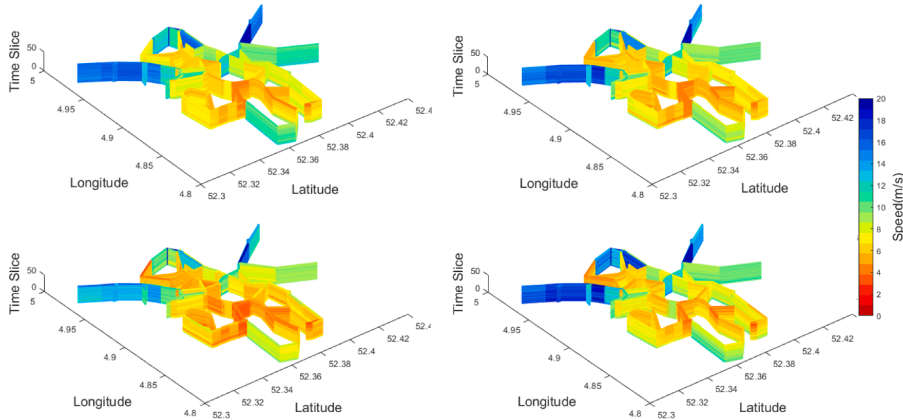


Fig. 9. Four shape-based speed models for Amsterdam.

4.2. Application II - Nationwide analysis

In this section, we present the results for the nationwide analysis. Since the speed limit of the road sections is available, we use the speed ratio instead of the absolute speed. Thus, a high ratio implies congestion (large speed drop) compared to low ratio. An example 3D speed ratio map of the entire dutch highway for a certain time period is shown in Fig. 14.

Since the speed limit is provided, we used $v^0 = 0.1$. Thus, all links with a speed ratio of less than 0.1 are considered as an uncongested region and hence are not clustered. We use $k = 5$ to cluster the congested links into different levels of congestion. We

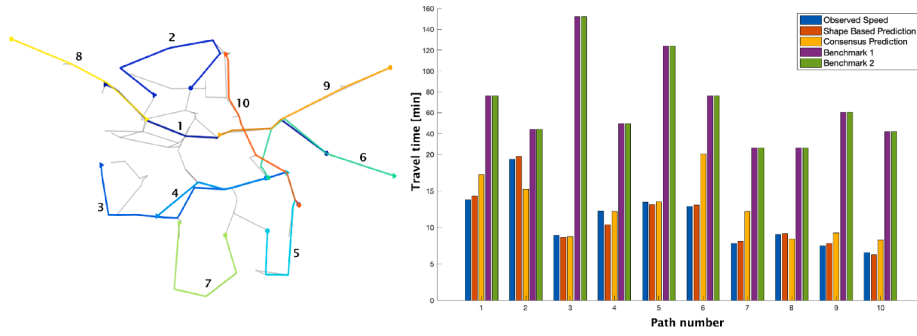


Fig. 10. The 10 routes used for validation with their corresponding travel time estimated from ground truth observed speed, predicted using the proposed shape-based method, the consensus method, and the two benchmark methods for a particular instance on an arbitrary day as an illustration.

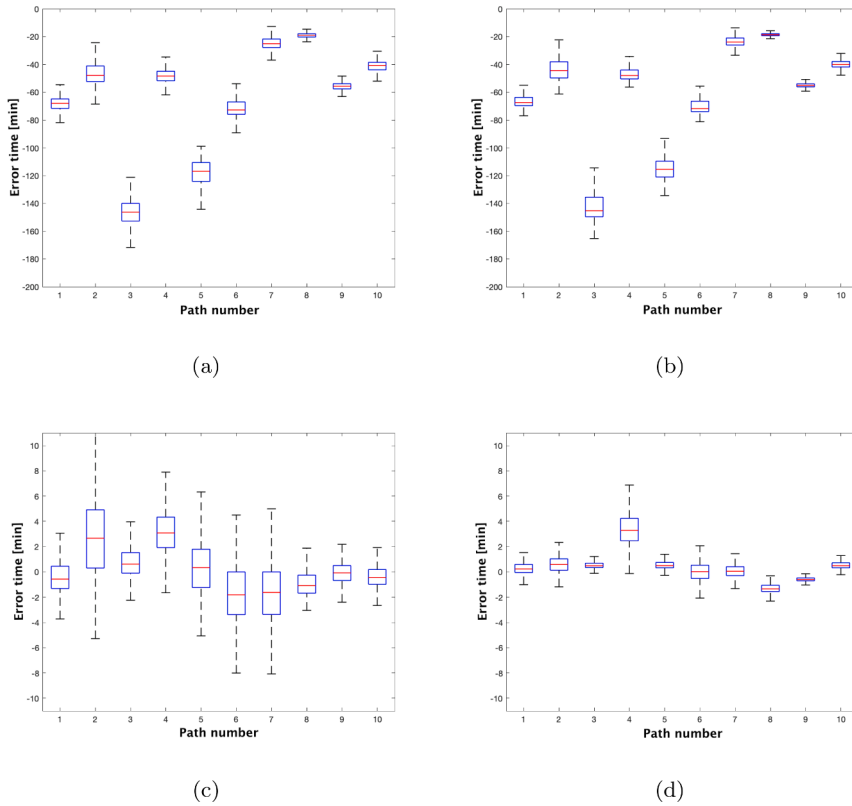


Fig. 11. Leave-one-out validation results for travel time prediction using the two benchmark methods, consensus models, and shape-based speed models for 35 days. The distribution of travel time error for the 10 routes within the 35 days for (a) historical average benchmark (b) data-driven historical average benchmark (c) consensus and (d) shape-based models.

identified the pocket of congestion from these clusters. An example of the top 10 pockets of congestion for a single day is shown in Fig. 15(a). The identified pockets of congestion is well-known bottlenecks within the Dutch highway network including the busy highway that connects Amsterdam to South Holland and Amsterdam to Utrecht. The corresponding 3D shapes of the pocket of congestion and the projected 2D shapes are shown in Fig. 15.

The projected shapes constitute the bag of shapes. In this case, again only 2 base shapes are needed for representing 95% of the variance within the bag of shapes. Thus, the bag of shapes is clustered into 2, in which the first base shape contains ≈ 4000 shapes and the second base shape ≈ 18000 shapes. The shape model of the first base shape has 16 principal components whereas the second is composed of 18 principal components. This shape model can be used to build a compact custom feature vector with a dimension of 20 per day, given $k = 5$ and the number of base shape = 2.

The feature vector of the 45 days can be clustered to reveal regularity between the days as outlined in Table 1. It includes the 4

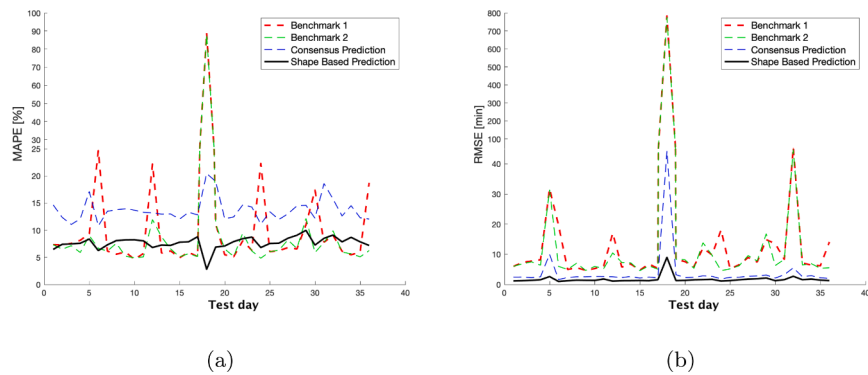


Fig. 12. (a) MAPE comparison results and (b) RMSE comparison results of the two benchmark methods, consensus and shape-based methods.

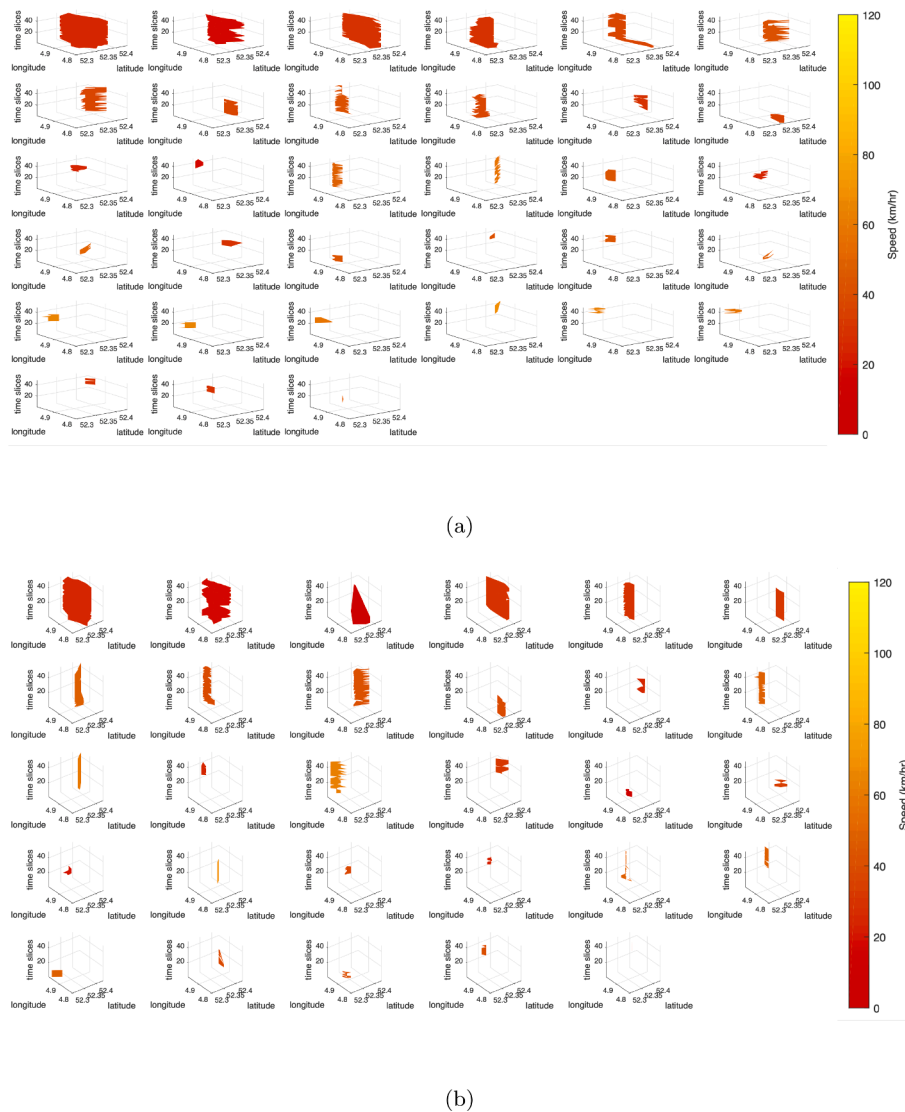


Fig. 13. 3D congestion shapes of two arbitrary days belonging to the same class. The congestions are ordered based on their spatial-temporal extent.

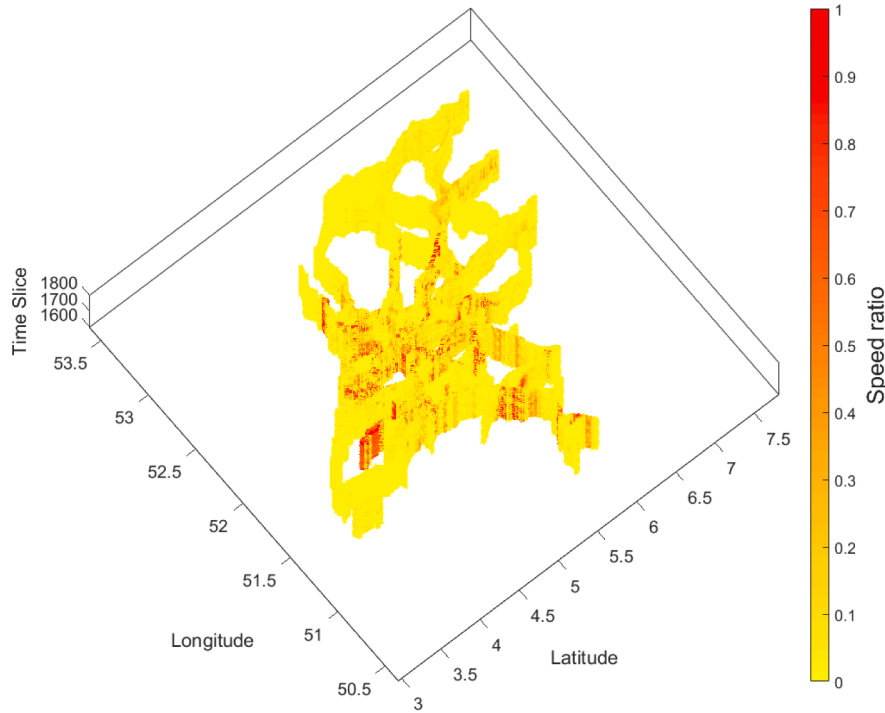


Fig. 14. 3D speed map of the entire dutch highway during the evening peak period from 16:30 to 18:00 (red implies congestion, yellow implies free-flow), which is ≈ 7 million speed ratio values.

shape-based speed models, the distribution of base shapes in the feature vector and the distribution of days in each class. The first 5 values of the feature vector dimension correspond to the occurrence of base shape 1 in each cluster k and the consecutive 5 values correspond to the frequency of the second base shape in each cluster. From the mean model of classes 1 and 4, we can see that they reflect relatively low congestion days which is in line with most of the weekdays categorized into these two classes. For all the classes, base shape 2 occurs more frequently than base shape 1. Only 1 to 2% of the network is congested (speed ratio > 0.1) in class 1 and 4 respectively whereas $\approx 41\%$ of the network is in congestion for classes 2 and 3.

We conduct a leave-one-out validation of the feature vectors to analyze if these daily variation are recurrent and hence, predictable. It achieves a prediction accuracy of 93% for the 45 days. A breakdown of the prediction is provided using a confusion matrix as shown in Table 2. Like in the Amsterdam case, the congestion classes seem to perform better than the relatively low congestion classes. One day from class 1 is wrongly classified as 4 and two day from class 4 are wrongly classified as either 2 or 3. This is attributed to not embedding information about the space and time extent of the pocket of congestion into the feature vector. With the current feature vector, we do not differentiate between large and small congestion conditions if they have the same shape. It will be classified into the same class.

Notwithstanding, the feature vector is still able to successfully distinguish between different classes. However, the usefulness of the classes heavily depends on the application. If the aim is to identify the evolution of different traffic states at the local level, we need to incorporate more information in the feature vector. The current feature vector provides only a global picture of the traffic dynamics. The feature vector can be further extended to incorporate spatial correlation of the base shapes for different parts of the network to provide more details at the local level. Furthermore, by incorporating the extent of the variation of each base shape in relation to the mean shape, we can account the extent of the congestion in the feature vector. However, the compact nature of the feature vector specified in this study allows incorporating more information about the network dynamics at a low computational cost.

Finally, we present the computational cost of building shape-based daily clusters for the entire Dutch network. The steps that differ between consensus and shape-based approach are the post-treatment and the model building steps. The post-treatment for the shape-based approach for a single day costs ≈ 20 min and processing the entire 45 days of data only takes around 15 h. However, we estimate that it would take months to apply the post-treatment methodology on a single day of data when using the consensus method. As for building the models, building the shape models took ≈ 11 hours. The main computational component in this is building the distance matrix between the bag of shapes to estimate the number of base shapes. However, this can easily be parallelized as these are just pairwise distances. In contrast, given that each day contains ≈ 97 million values, running OEM (one element move) and optimizing over 45 days for the consensus method, the computational complexity is in the order of magnitude of months and not hours. Hence, we do not compute the consensus model for the nationwide analysis. In short, processing 45 days of data for the entire Dutch highway network using our approach costs less than 2 days on a 64-bit machine without any parallelization, which is very promising for large-scale network dynamics analysis. This allows conducting in-depth analysis and interpretation of these shapes and extending the feature

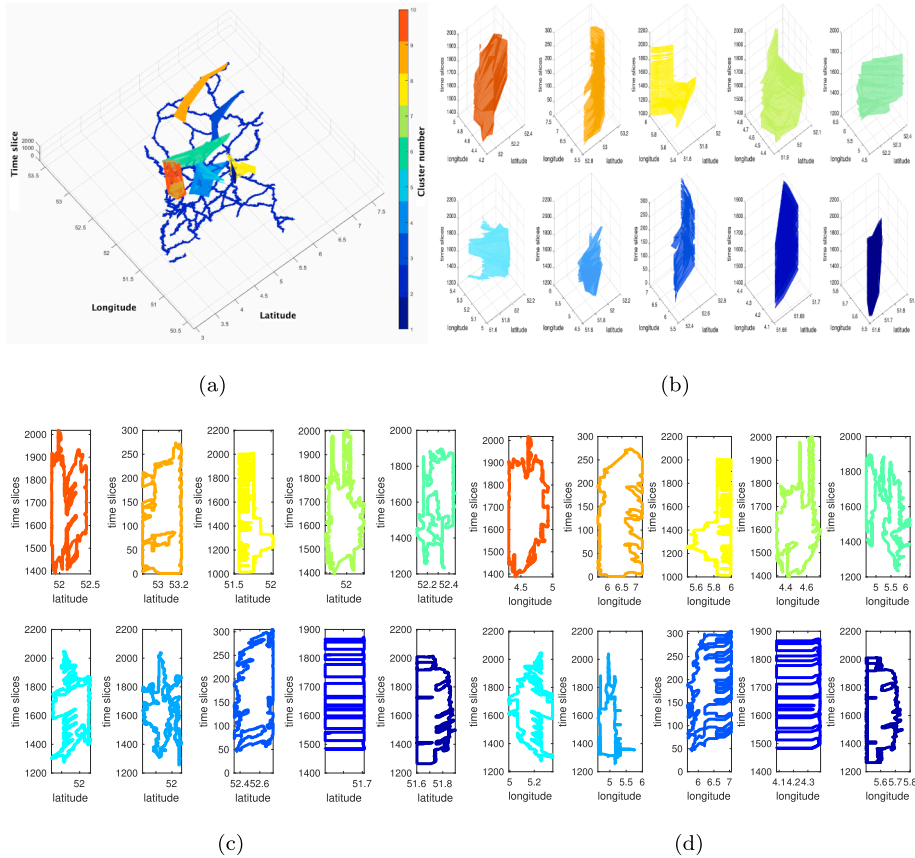


Fig. 15. Representing the top pockets of congestion for a given day as 2D shapes of the entire Dutch highway network (a) top 10 pockets of congestion for that day (b) corresponding 10 3D shapes (c) projected 2D shape representing the boundary between latitude and time (d) projected 2D shape representing the boundary between longitude and time. Note that the range of values shown in the axes in (b), (c) and (d) varies due to the study area size.

vector to further understand the different aspects of a network exhibiting regular patterns.

5. Conclusion

In this paper, we propose an efficient method for representing the traffic dynamics of a large-scale network using shapes. To this end, we first identify the pockets of congestion (i.e. connected sub-networks with low speed over space and time) and extract 3D shapes from these pockets. These 3D shapes are used to build a representative bag of shapes for the network. This bag essentially contains all the different shapes of congestion that occurs within this network. We extract the essence of these shapes as base shapes and build a Statistical Shape Model (SSM) for each of them. The SSM combined with the traffic variable is used to define custom feature vector for a daily pattern. These feature vectors are then used for accurate network travel time predictions by clustering feature vectors for a historical data set, and then matching the prevailing traffic conditions to one of the historical clusters. We also show how these feature vectors can be used to classify a new daily pattern into one of the historical clusters, which indicates the representativeness of the historical cluster and the feature vector.

We demonstrate our approach on an urban network (Amsterdam) and on a large-scale network (entire Dutch highway network) with promising results. We achieve a mean travel time prediction accuracy of 8%, which is $\approx 44\%$ improvement compared to the consensus method for the Amsterdam network. We are able to achieve this improvement with just two base shapes and a feature vector of dimension 40, which is a 99% data dimensionality reduction. For the nationwide analysis, we use the shape-based feature vectors to reveal the regularity between the days. The clustering provides meaningful network patterns with different types of congestion - low, mild, heavy congestion patterns. There are also significant differences between weekday and weekend patterns, with a majority of the weekends clustered under low to mild congestion classes, except for a few exemptions. These network pattern clusters are then used to build a classifier, which is able to achieve a classification accuracy of 93%. This implies that the feature vector is able to generalise and successfully uncover the difference between the daily patterns contained in the historical data set.

From the results of the two networks, we draw the following conclusions. We conclude that using shapes for understanding network traffic dynamics offers an efficient, insightful and accurate method. We have shown that defining pockets of congestion in a traffic

Table 1
Description of Dutch highway network traffic states.

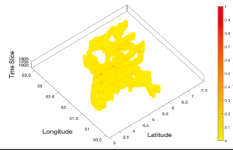
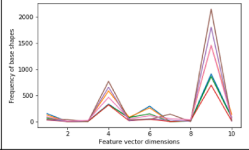
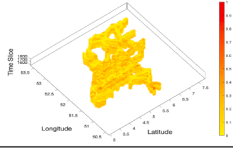
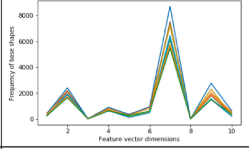
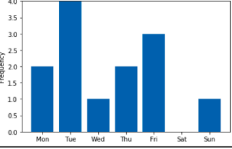
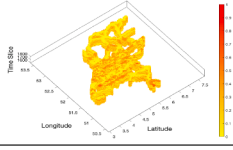
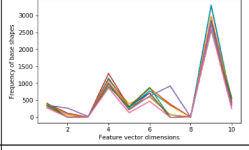
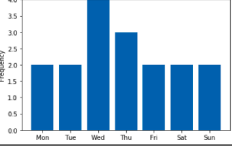
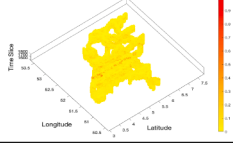
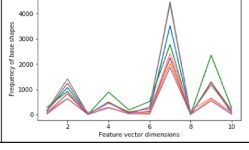
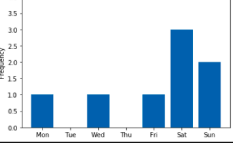
Class	Mean model	Distribution of base shapes in each class	Distribution of days in each class
1			
2			
3			
4			

Table 2
Leave-one-out classification results.

KNOWN	PREDICTED			
	Class 1	Class 2	Class 3	Class 4
Class 1	0.86	0	0	0.14
Class 2	0	1	0	0
Class 3	0	0	1	0
Class 4	0	0.08	0.07	0.75

network using shapes can reveal regularity between the daily patterns. The compact nature of the feature vector coupled with using tangible attributes allows to generate interpretable outcomes. We also found that for both case study networks the feature vector is able to encompass the congestion dynamics better than the free flow dynamics. In the first application, the travel times were underestimated and in the second application, the relatively low congestion days were the ones that were classified wrongly.

We postulate that there are many possible paths to further explore, refine, and improve the properties of the method. Here, we present key directions for further research. First, the feature vector can be extended to enrich it as well as make it more interpretable by including information about the size of the shape (which corresponds to the extent of congestion) or location as to where the shape occurred within a network. Moreover, the time dimension can be incorporated into the feature vector so that one can differentiate between the occurrence of certain congestion shapes in the morning and evening period and thus the feature vector can provide more details about the congestion propagation. Second, the evolution of shapes in different regions can be used to analyse local congestion propagation. This will allow associating certain shapes to certain bottlenecks and correlate the bottleneck with other bottlenecks of the network. Third, more sophisticated representative speed maps for each class can be built, other than just the mean model. Since, each day is represented by a set of shapes, we can consider the problem as a jigsaw puzzle of shapes and solve it accordingly. Finally, we can interpret these congestion shapes in relation to theoretical traffic flow studies and explore the influence of network structure on these shapes. By decomposing the network dynamics problem into a set of shapes, the well-studied problem can be re-imagined as a vision problem, which is compelling for humans to grasp and it also opens up many exciting avenues of future research.

CRediT authorship contribution statement

P. Krishnakumari: Conceptualization, Methodology, Software, Visualization, Writing - original draft. **O. Cats:** Conceptualization, Supervision, Writing - review & editing. **H. van Lint:** Conceptualization, Funding Acquisition, Supervision, Writing - review & editing.

Acknowledgements

This research is supported by the SETA project funded by the European Union's Horizon 2020 Research and Innovation program under the grant agreement No 688082, and by the NWO/TTW project MiRRORS under grant agreement 16270. We thank the anonymous reviewers for the constructive feedback which has greatly improved this paper.

References

- Brandt, A., Ron, D., 2013. Multigrid solvers and multilevel optimization strategies. *Multilevel Optimiz. VLSICAD* 14, 1.
- Chevalier, C., Saffro, I., 2009. Comparison of coarsening schemes for multilevel graph partitioning. In: *International Conference on Learning and Intelligent Optimization*. Springer, pp. 191–205.
- Coates, T.F., Taylor, C.J., Cooper, D.H., Graham, J., 1995. Active shape models-their training and application. *Comput. Vision Image Understand.* 61 (1), 38–59.
- Cover, T.M., Thomas, J.A., 2012. *Elements of Information Theory*. John Wiley & Sons.
- Dörnfelder, M., Guo, J., Komusiewicz, C., Weller, M., 2014. On the parameterized complexity of consensus clustering. *Theoret. Comput. Sci.* 542, 71–82.
- Fawcett, T., 2006. An introduction to roc analysis. *Pattern Recogn. Lett.* 27 (8), 861–874.
- Filkov, V., Skiena, S., 2004. Integrating microarray data by consensus clustering. *Int. J. Artif. Intell. Tools* 13 (04), 863–880.
- Gazis, D.C., Herman, R., Potts, R.B., 1959. Car-following theory of steady-state traffic flow. *Oper. Res.* 7 (4), 499–505.
- Ge, Q., Fukuda, D., 2019. A macroscopic dynamic network loading model for multiple-reservoir system. *Transp. Res. Part B: Methodol.* 126, 502–527.
- Geroliminis, N., Daganzo, C.F., 2008. Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings. *Transp. Res. Part B: Methodol.* 42 (9), 759–770.
- Gipps, P.G., 1981. A behavioural car-following model for computer simulation. *Transp. Res. Part B: Methodol.* 15 (2), 105–111.
- Godfrey, J., 1969. The mechanism of a road network. *Traffic Eng. Control* 8 (8).
- Gu, Z., Saberi, M., 2019. A bi-partitioning approach to congestion pattern recognition in a congested monocentric city. *Transp. Res. Part C: Emerg. Technol.* 109, 305–320.
- Haddad, J., Ramezani, M., Geroliminis, N., 2013. Cooperative traffic control of a mixed network with two urban regions and a freeway. *Transp. Res. Part B: Methodol.* 54, 17–36.
- Hamedmoghadam, H., Ramezani, M., Saberi, M., 2019. Revealing latent characteristics of mobility networks with coarse-graining. *Sci. Rep.* 9 (1), 1–10.
- Helbing, D., Treiber, M., 1998. Gas-kinetic-based traffic model explaining observed hysteretic phase transition. *Phys. Rev. Lett.* 81 (14), 3042.
- Jolliffe, I., 2011. *Principal Component Analysis*. Springer.
- Kendall, D.G., 1989. A survey of the statistical theory of shape. *Stat. Sci.* 87–99.
- Kerner, B.S., Rehborn, H., Aleksic, M., Haug, A., 2004. Recognition and tracking of spatial-temporal congested traffic patterns on freeways. *Transp. Res. Part C: Emerg. Technol.* 12 (5), 369–400.
- Knoop, V.L., Van Lint, H., Hoogendoorn, S.P., 2015. Traffic dynamics: Its impact on the macroscopic fundamental diagram. *Physica A* 438, 236–250.
- Krishnakumari, P., Cats, O., van Lint, H., 2019. Heuristic coarsening for generating multiscale transport networks. *IEEE Trans. Intell. Transp. Syst.*
- Krishnakumari, P., Nguyen, T., Heydenrijk-Ottens, L., Vu, H.L., van Lint, H., 2017. Traffic congestion pattern classification using multiclass active shape models. *Transp. Res. Rec. J. Transp. Res. Board* (2645), 94–103.
- Leclercq, L., Chiabaut, N., Trinquier, B., 2014. Macroscopic fundamental diagrams: A cross-comparison of estimation methods. *Transp. Res. Part B: Methodol.* 62, 1–12.
- Lighthill, M.J., Whitham, G.B., 1955. On kinematic waves ii. a theory of traffic flow on long crowded roads. *Proc. R. Soc. Lond. Ser. A. Math. Phys. Sci.* 229 (1178), 317–345.
- Lopez, C., Krishnakumari, P., Leclercq, L., Chiabaut, N., Van Lint, H., 2017a. Spatiotemporal partitioning of transportation network using travel time data. *Transp. Res. Rec.* 2623 (1), 98–107.
- Lopez, C., Leclercq, L., Krishnakumari, P., Chiabaut, N., Van Lint, H., 2017b. Revealing the day-to-day regularity of urban congestion patterns with 3d speed maps. *Sci. Rep.* 7 (1), 14029.
- MacQueen, J. et al., 1967. Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, Oakland, CA, USA, pp. 281–297.
- Mahmassani, H., Williams, J.C., Herman, R., 1987. Performance of urban traffic networks. In: *Proceedings of the 10th International Symposium on Transportation and Traffic Theory*. Elsevier, Amsterdam, The Netherlands, pp. 1–20.
- Nagel, K., Schreckenberg, M., 1992. A cellular automaton model for freeway traffic. *J. Phys. I* 2 (12), 2221–2229.
- Nwb (nationaal wegenbestand), <https://www.arcgis.com/home/item.html?id=36486316f8674b3fa15c9ee2b2d8ecd7#data>.
- Opensource heuristic coarsening code, <https://github.com/Panchamy/Heuristic-Coarsening/wiki>.
- Payne, H.J., 1971. Model of freeway traffic and control. *Math. Model Public Syst.* 51–61.
- Prigogine, I., Herman, R., 1971. Kinetic theory of vehicular traffic. *Tech. rep.*
- Saeedmanesh, M., Geroliminis, N., 2016. Clustering of heterogeneous networks with directional flows based on "snake" similarities. *Transp. Res. Part B: Methodol.* 91, 250–269.
- Saffro, I., Sanders, P., Schulz, C., 2015. Advanced coarsening schemes for graph partitioning. *J. Exp. Algorithmics (JEA)* 19, 2.
- Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis Mach. Intell.* 22 (8), 888–905.
- Tarjan, R., 1972. Depth-first search and linear graph algorithms. *SIAM J. Comput.* 1 (2), 146–160.
- Treiber, M., Helbing, D., 2002. Reconstructing the spatio-temporal traffic dynamics from stationary detector data. *Cooperative Transp. Dyn.* 1 (3), 1–3.
- Whitham, G., 1975. *Linear and Nonlinear Waves*. Modern Book Incorporated.
- Yildirimoglu, M., Ramezani, M., Geroliminis, N., 2015. Equilibrium analysis and route guidance in large-scale networks with mfd dynamics. *Transp. Res. Procedia* 9, 185–204.