Time–Space Diagram Revisited

Afian Anwar, Wei Zeng, and Stefan Müller Arisona

Widely used in the design and analysis of transportation systems, time–space diagrams were developed in an era of data scarcity, when it was necessary to obtain data by means of driver logs, human observers, and aerial photographs. This paper shows how time–space diagrams remain relevant today, in an era of data abundance. An application efficiently encodes the trajectories of bus GPS data in a time–space cube and uses simple geometric methods to calculate and to visualize the headways and separation of buses on a bus route. These methods are discussed in detail. How they can be used as the basis of a software package that monitors performance measures for a variety of applications is explored.

Time–space diagrams are commonly used to solve a wide variety of transportation problems. Typically, the variable x denotes the distance traveled along a guideway from some arbitrary reference point, and t denotes the time elapsed from an arbitrary instant (*1,* pp. 1–2). The study of how the position of a vehicle changes over time leads to a better understanding of the performance characteristics of the transportation system under analysis.

The biggest challenge in the use of a time–space diagram is collecting sufficiently accurate data to build one (*1,* pp. 12–13). Time–space diagrams came into prominence in the 1950s and 1960s, when transportation engineering was in its infancy. Back then, data collection was tedious and expensive. Today, it is standard practice for transport operators to know where all their vehicles are at all times. For example, bus operators routinely use bus location data to inform passengers when the next bus is arriving, and taxi operators use real-time taxi GPS data in their booking systems.

All these data are location based and contain at minimum a position (latitude and longitude) and time stamp. This paper proposes the use of such data to build time–space diagrams for the analysis and visualization of transportation operations. The main contributions of this paper are as follows:

- A way to store vehicle trajectories efficiently as line segments in three-dimensional space for fast retrieval,
- Simple geometric methods inspired by the time–space diagram that apply intersection tests with these line segments to find vehicle headways and spacing,
- A visualization and analysis tool that can help transit operators and regulators better understand network performance and behavior, and
- A discussion of how any transportation system with big data capability can use time–space diagrams to monitor service levels, network capacity, and delays.

A. Anwar, Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139. W. Zeng, Future Cities Laboratory, Department of Architecture, ETH Zurich, 8092 Zurich, Switzerland. S. M. Arisona, Institute of 4-D Technologies, University of Applied Sciences and Arts, Northwestern Switzerland FHNW, Bahnhofstrasse 6, 5210 Windisch, Switzerland. Corresponding author: A. Anwar, afian@csail.mit.edu.

Transportation Research Record: Journal of the Transportation Research Board, No. 2442, Transportation Research Board of the National Academies, Washington, D.C., 2014, pp. 1–7.
DOI: 10.3141/2442-01

RELATED WORK

Time–Space Diagrams

Time–space diagrams study how vehicles move in time and are regularly used to determine the maximum throughput of airport runways (*2,* pp. 408–409), to devise optimal train schedules (*3,* pp. 55–56), and to coordinate traffic lights (*4*).

It is convenient to plot x on the x-axis and t on the y-axis as in Figure 1, which shows the trajectories of three vehicles, a, b, and c, on a straight road. By plotting the position of vehicles along a guideway over time to obtain the function $x(t)$, the displacement x of the vehicle for every time t, one can easily deduce headways (when viewed from a fixed position) and separation (when viewed from a fixed point in time). The inverse of the slope of each trajectory gives the vehicle's speed, and its curvature gives acceleration. An increasing slope indicates that the vehicle is decelerating; a decreasing slope shows that the vehicle is accelerating. As noted by Daganzo, a vertical line through the diagram identifies the times at which successive vehicles pass a stationary observer, and a horizontal line identifies the vehicle positions at the given time (*1,* pp. 13–14). The times between consecutive vehicle observations at a fixed location are called headways (h_{bc} in Figure 1), and the distance separation between consecutive vehicles at a given instant is called spacings (s_{bc} in Figure 1).

Time–Space Cubes

So far, the time–space diagram and methods of analysis are limited to two dimensions, time t and distance x. Adding a third dimension y to represent the distance traveled in the y direction gives a time–space cube. Time–space cubes were first developed by Hägerstrand to model the time geography of individual movement, and they have been used primarily by urban planners and for understanding the social interactions of people and place. In its typical form, the cube has on its base a representation of geography, for example, a city map, and the cube's height represents time (z-axis) (*5*).

As in the time–space diagram, trajectories are represented by joining together x, y, and t points belonging to the same vehicle, so that as time progresses, the trajectory moves upward in time. Each vertical slice of the time–space cube along the vehicle's path shows its trajectory in two dimensions, illustrating that the time–space cube is a complete summary of the progress of a vehicle as captured by the time–space diagram (Figure 2). The trajectory of the bus through

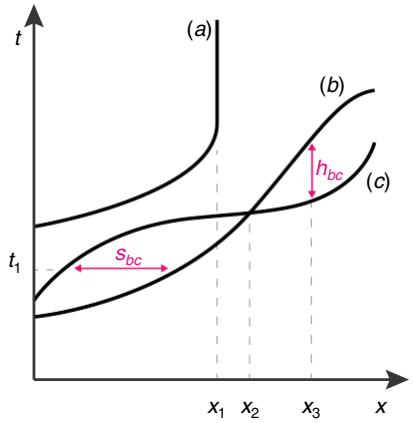


FIGURE 1 Time–space diagram of vehicles *a*, *b*, and *c*. Vertical line for vehicle *a* at position x_1 shows that vehicle has come to complete stop. When trajectories intersect, as vehicle *c* does with vehicle *b* at position x_2, vehicle *c* has overtaken vehicle *b*. Horizontal distance s_{bc} between trajectories *b* and *c* at time t_1 gives spacing; vertical distance h_{bc} between trajectories *b* and *c* at position x_3 gives headway.

time–space is represented by the solid gray line; its actual path can be found by projecting its time–space path onto the two-dimensional base of the cube.

The work in this paper extends earlier work by first using line segments in a time–space cube to represent the observed trajectories of buses in Singapore and then making vertical and horizontal cuts on the cube's constituent time–space diagrams to efficiently find vehicle headway and spacing. Unlike previous work that used headway and spacing data collected at selected geographic locations (*6*), the approach in this study allows performance metrics to be calculated at any arbitrary point on the bus route, showing how they vary over time.

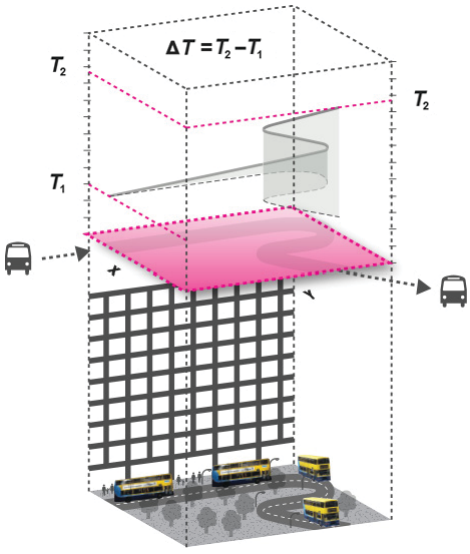


FIGURE 2 Bus enters time–space cube at time T_1 and leaves at time T_2. The time spent in cube is given by $T_2 - T_1$.

OUTLINE

First, the motivation for the approach and the data used for the study are described. Then the problem is formulated and notation defined, and a way to efficiently store and query the data with a spatial index is offered. Next it is shown how the methods, packaged in an application (app), could be used to solve the real-world problem of bus bunching by allowing users to compute and visualize bus headways and spacing easily and intuitively. Finally, how the work in this study could be used in any transportation system with big data capability to monitor service levels, network capacity, and delays is discussed.

MOTIVATION

Transport regulators are often held accountable for the level of service and reliability of the transport operators under their purview. In Singapore, the Land Transport Authority (LTA) plans routes and establishes minimum service standards for bus lines operated by the Singapore Bus Service and Singapore Mass Rapid Transit. LTA is responsible for both ensuring that these minimum service standards are met and investigating complaints from the general public. Meeting this responsibility traditionally has required the use of observers to survey bus passenger loads, frequency, and reliability, an expensive and time-consuming task. The work in this paper shows how big data can be used to replace these surveys with a visual analysis tool that uses time–space diagrams to capture the performance metrics for an entire bus network with greater accuracy and lower cost and at scale.

DATA

The bus data set (2 GB of text-encoded trip data) used in the study was made available through a recent initiative by LTA to monitor bus ridership. It contains 3 months (June, July, and August 2013) of bus GPS and passenger load records for each of Singapore's 400 bus lines, collected automatically by LTA's real-time passenger information system.

Each record contains the time stamp, bus route ID, bus ID, bus stop ID of the current bus stop, direction (inbound or outbound), maximum capacity, and number of passengers for each bus. Records are logged each time the bus arrives at a bus stop, which allows the exact position of each active-duty bus to be tracked over the course of a day. Because most passengers in Singapore use smart cards when getting on and off buses (*7*), approximately how many people are on board can be known. Each record contains the unique ID of the bus that it belongs to and the bus stop ID of the bus stop where the bus is, and this information can be cross-referenced with an index of the GPS locations of each bus stop to reproduce the bus's trajectory.

PROBLEM FORMULATION

In this section the problem is formulated, notation is defined, and an efficient method is proposed for storing and querying bus data. Consider the task of monitoring service levels by finding mean bus

headway and spacing at any location on the bus route over some time interval. A trajectory consists of points $p_1, p_2, \ldots, p_n$, where each point p is a 3-tuple consisting of a longitude x, a latitude y, and a time stamp t as observed by the automatic vehicle location system at location (x, y) at time t. Since each trajectory consists of a path in time–space, it is natural to represent the trajectory as a series of ordered line segments $l_1, l_2, \ldots, l_{n-1}$ in a time–space cube. Each line segment l_i connects successive points p_i and p_{i+1}, where $p_i.t \leq p_{i+1}.t$; $\forall i \in \{1, 2, \ldots, n-1\}$. For a large number of trajectories, iterating through every line segment in each trajectory to perform an intersection test is not practical. Because the goal is to query these trajectories quickly, a better approach is needed.

Spatial Index

The solution is to store line segments in an R-tree, a data structure similar to a balanced binary search tree that is commonly used to index geometric objects such as points, lines, and polygons (*8–11*). The R-tree data structure groups nearby trajectories and represents them with the minimum bounding rectangle that completely contains these trajectories in the next higher level of the tree. Queries on R-trees are fast because these bounding rectangles can be used in the decision of whether to search inside the subtree. In this way, most of the nodes in the tree are never read during a search.

For each line segment l_i, an axis-aligned bounding box (the smallest box that contains all the points on the line) is built that has as its start and end vertices points p_i and p_{i+1}. Then line segments are added to the R-tree one at a time so that the bounding box of each parent node in the R-tree completely contains the bounding boxes of its children. To keep the tree balanced, nodes are always added to the subtree that requires the least enlargement of its bounding box. This guarantees that one can search for line segments in $O(\log(n))$ time (*12*). It takes on average 250 ms on a laptop with a 2.4-GHz dual-core Intel Core i5 processor to add every trip made on a single bus line (both inbound and outbound for 1 day) to the spatial index. This is fast enough for the study's purposes because the app loads new data from other days or bus lines only when requested by the user.

Other methods for indexing line segments are possible—for example, by incoming and outgoing bus stops. However, that method is more computationally expensive because it requires keeping a reference to each trajectory that enters and leaves every bus stop, as well as finding the specific pair of bus stops that the line segment cuts.

Headways and Spacing

The time between successive buses gives the headway and the distance, the spacing. Consider two successive trajectories a and b. On a time–space diagram with x on the x-axis and t on the y-axis, the headway at $x = x'$ $[h_{ab}(x')]$ can be found by measuring the distance between the intersection of a and b with the vertical line at some fixed x'. Similarly, the spacing at $t = t'$ $[s_{ab}(t')]$ can be found by measuring the distance between the intersection of a and b with the horizontal line at some fixed t'.

When this analysis is extended to the time–space cube, the intersection of these trajectories with the $x - y$ plane at certain time $t = t'$ gives the location of the bus at t'. If this plane is superimposed on a road segment, these intersections give the position of the buses at t', which is analogous to taking an aerial photograph of that road segment. The horizontal distance between each intersection point gives the separation between each bus. Similarly, the vertical distance between the intersections of these trajectories with the $y - t$ plane at a fixed point $x = x'$ perpendicular to the road gives the headway between buses. This is analogous to an observer standing on the side of the road at x' and recording the time between successive buses.

The preceding analysis produces a geometric interpretation for an algorithm with which to calculate headways and spacing. The mean headway is the average vertical separation of the intersection between a set of trajectories and a vertical rectangle; the mean spacing is then the average horizontal separation of the intersection between a set of trajectories and a horizontal rectangle (Figure 3). To find these

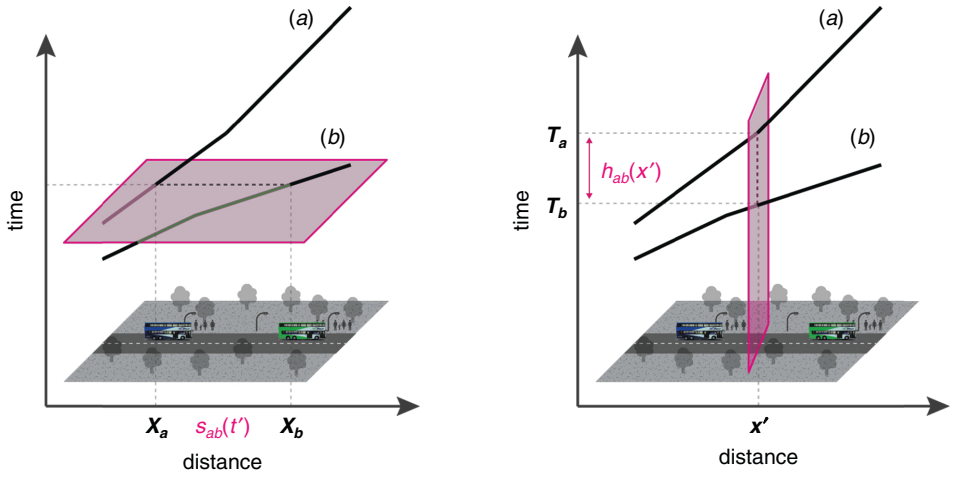


FIGURE 3 Bus headways and separation. Bus *b* is ahead of bus *a* on straight road segment. Vertical line at given abscissa identifies vehicles at given time (represented by horizontal plane), and horizontal line at specified ordinate identifies time at which vehicle passes location (represented by vertical plane).

intersections efficiently, first use the spatial index to check which line segments have bounding boxes that intersect the rectangle. Next, run a simple line and polygon intersection test to find the exact point of intersection, if any. The spatial index search is $O(\log(n))$, and the intersection test is $O(1)$, so the algorithm runs in $O(\log(n))$ time. The average execution time for the entire operation is 10 ms, which is fast enough for it to be run in real time, that is, with negligible wait time or lag.

NETWORK VISUALIZATION AND ANALYSIS APP

A visualization and analysis app (Figure 4) that automatically parses and represents bus data as trajectories in a time–space cube was developed for exploring whether use of the time–space diagram with big data could offer a better understanding of transportation network performance and behavior. This app, intended for use by transit operators and regulators, allows users to manage and visualize performance metrics such as headways and spacing in an interactive and easily understandable way. A possible application of the software is for the diagnosis of bus bunching, that is, when two or more buses that were scheduled to be evenly spaced on the same route run at the same time. Bunching is characterized by irregular headways and spacing and often results in longer wait times for riders, bus overcrowding, and an overall decrease in service level and capacity (*13*).

Concern about bus bunching is not academic. Transport regulators such as the LTA take bunching seriously and have introduced incentives and penalties to encourage bus operators to improve reliability and reduce overcrowding. Under a new framework, bus operators in Singapore are required to reduce bus bunching by asking bus drivers to speed up or slow down or by adding buses midstream (*14*).

Bus Bunching

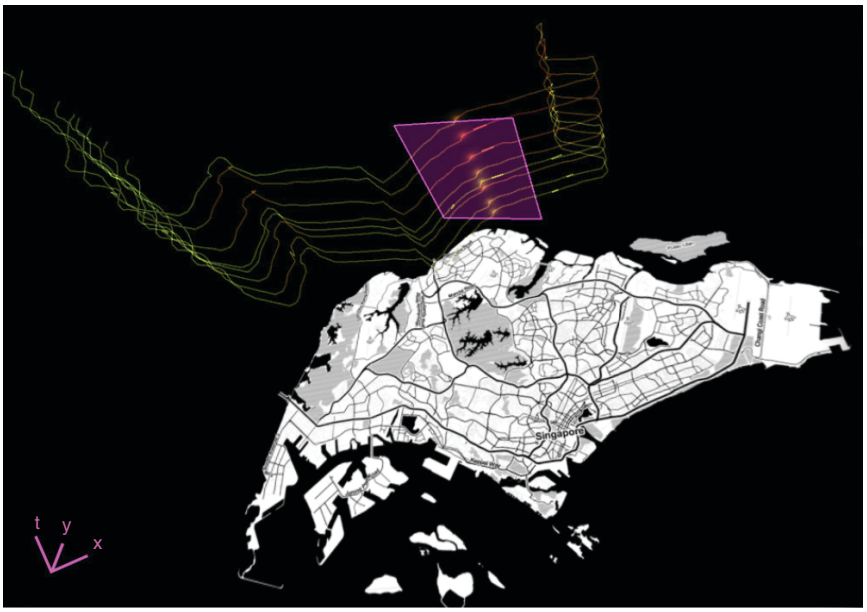
Bus bunching arises naturally because of the stochastic nature of passenger arrivals at bus stops. If more than the average number of passengers are waiting at a stop for an arriving bus, this bus will likely be delayed (*15*). The situation worsens when the delayed bus picks up passengers that should have been picked up by the bus behind it; eventually the following bus catches up and the two buses end up traveling as a unit (*16*).

There has been significant research interest in designing control mechanisms to reduce bus bunching (*17–19*), but before such control systems can be implemented, the bus operator must be able to quantify the severity of the problem. This is typically done with a traditional time–space diagram that plots the bus's position along the route on the *y*-axis and plots time on the *x*-axis. However, this approach abstracts the bunching problem away from geography. The proposed app described in the next section allows the user to easily identify and visualize bus bunching events temporally and spatially at any arbitrary point along a bus route and to zoom in on problem areas.

Interface Design and User Interaction

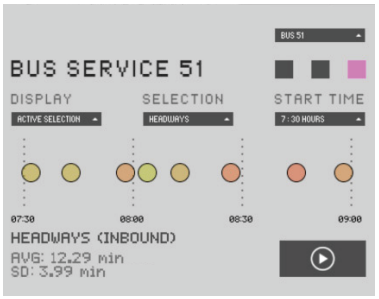
The app was prototyped in Java primarily so it could be built on previous work on visualizing large spatiotemporal data sets (*20*).

Two panels are displayed on the main screen of the app. The left panel is the time–space cube view (Figure 4*a*), which shows bus trajectories from a single bus line, Bus 51, plotted in three-dimensional space. Bus 51 was chosen because of its importance and popularity. It is a cross-island bus line that starts in a residential neighborhood



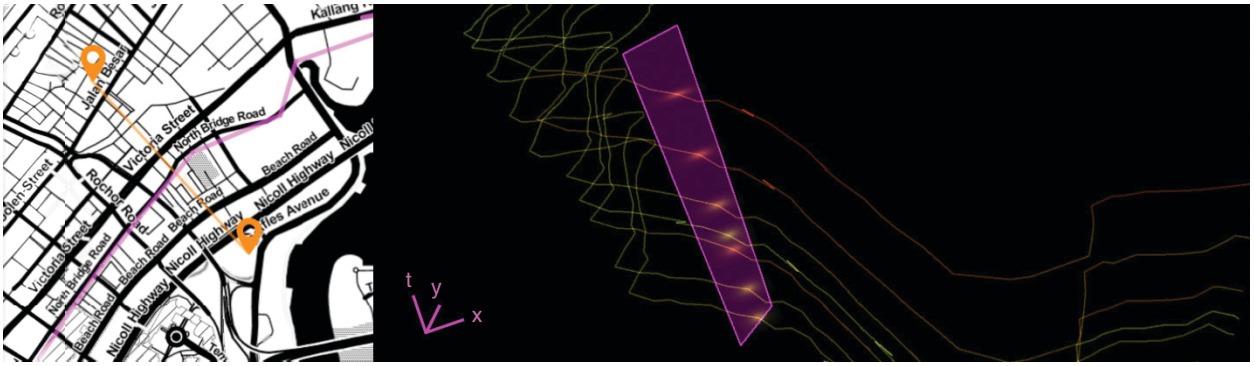
(a)

(b)



(c)

FIGURE 4 Network visualization and analysis app interface: (*a*) time–space cube view, (*b*) map selection view, and (*c*) visualization options view (avg = average; SD = standard deviation).



(a) (b)

FIGURE 5 Average headway at specific point on route: (*a*) user drops two markers (in orange) across bus route (in light pink) to specify location where bus headways are to be calculated and (*b*) resultant vertical time slice is then visualized in three-dimensional space together with intersecting trajectories.

in the west, enters the city center, and ends in the eastern part of the country. The base of the cube is a city map of Singapore, the bus route highlighted in bright pink. Trajectories are colored according to how full each bus is: green for empty and red for full. The *x*-axis, *y*-axis, and *z*-axis correspond to longitude, latitude, and time.

The right panel is the map selection view (Figure 4*b*), which allows users to drop markers on a zoomable map to define the plane that will be used in the intersection tests. The visualization options view (Figure 4*c*) lets users choose via a drop-down menu the day, route number, and route direction (inbound or outbound), the type of statistics (headway or spacing), and the time period (00:00 to 24:00).

Headways

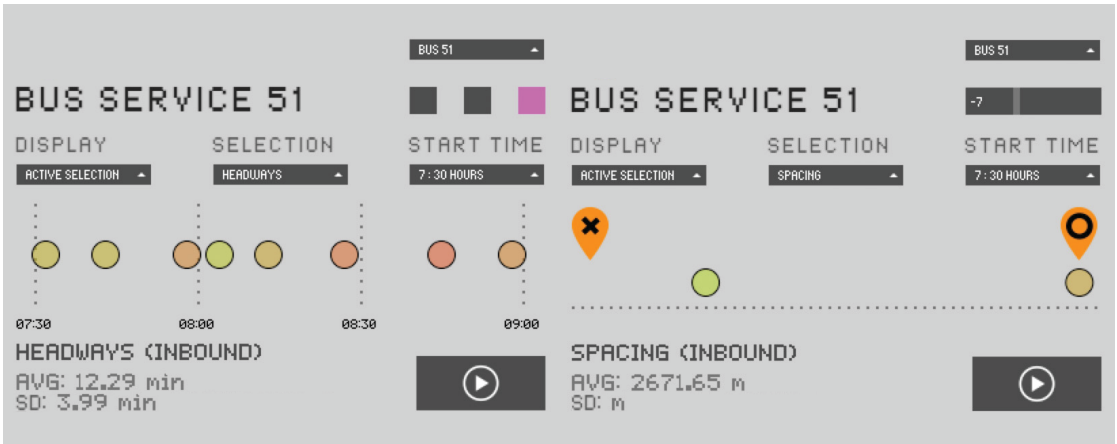
To find the average headway at a specific point on the route for a specified time interval, the user drops markers on the map to draw a line segment across the road before selecting the start time and the duration of the interval (Figure 5*a*). This reinforces the notion that the user is making a cut through time across the road.

Pressing the run button generates a vertical time slice, and the app retrieves a list of intersecting trajectories and their intersection points. It then calculates both the average headway and the standard deviation and displays a chart (Figure 6*a*) showing how buses (green for empty, red for full) passing by the line segment are separated in time.

This is equivalent to recording the times at which buses pass by a stationary observer. Each bus is represented by a circle, and overlapping circles indicate bus bunching. Simultaneously, the time–space cube view zooms in on the time slice and highlights intersecting trajectories with a dull glow (Figure 5*b*), allowing the user to visualize how trajectories intersect the vertical slice in time–space. As with the bus trajectory, the color of the glow conveys the number of passengers on board.

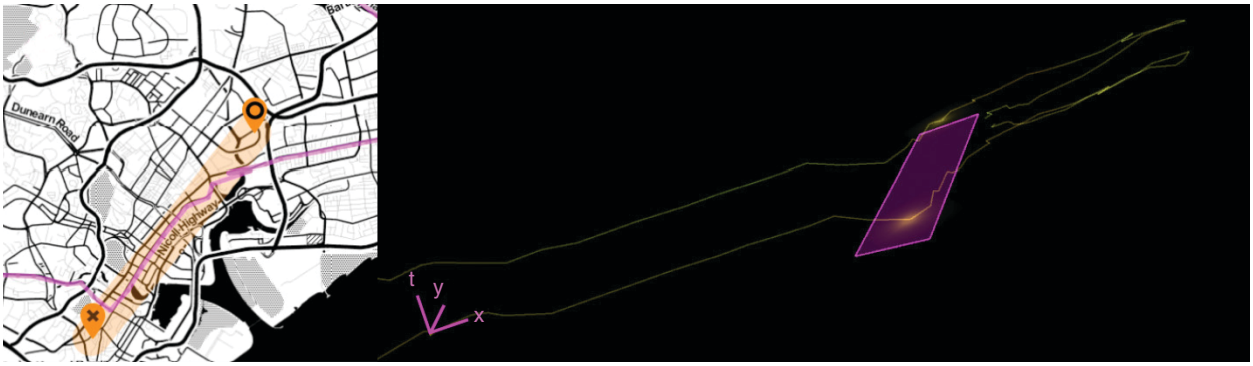
Spacing

To find the average spacing of vehicles along a road segment, the user drops two markers (an X and an O) at the start and end points



(a) (b)

FIGURE 6 App calculates average headway and spacing and their standard deviations and plots bus's relative positions in (*a*) time and (*b*) space. Overlapping circles indicate bus bunching. Bus's passenger load is encoded in color of each circle.



(a) (b)

FIGURE 7 Average spacing of vehicles along road segment: (*a*) user drops X and O markers along road segment where bus spacings are to be calculated and (*b*) resultant horizontal time slice is then visualized in three-dimensional space together with intersecting trajectories.

(Figure 7*a*). This generates a rectangular bounding box that defines the horizontal time slice used to find intersecting trajectories.

The user then selects the start time and presses the run button to retrieve a list of intersecting trajectories and their intersection points. The app calculates the average spacing and standard deviation and displays the results in a chart (Figure 6*b*). This is equivalent to an aerial photograph (represented by the light orange area in Figure 7*a*) of the road segment taken at a point in time. Again, each bus is represented by a circle, and overlapping circles indicate bus bunching.

The user can adjust a time slider to move the horizontal time slice up or down (forward and backward in time) to simulate the effect of watching a time-lapse video of buses moving along the road segment. Again, the time–space cube view zooms in on the time slice and highlights intersecting trajectories with a dull glow (Figure 7*b*).

DISCUSSION OF RESULTS

Indexing real-world bus trajectories as line segments in a time–space cube and the use of intersection tests between these trajectories and rectangle time slices to efficiently calculate headways and spacing have shown that the classic time–space diagram remains an important tool in transportation system analysis. Although this work was motivated by the problem of deriving performance metrics for a bus network, the methods presented in this paper are general and so can be applied in contexts other than public transit.

For example, since the real-time GPS positions of all aircraft in North America and Europe are logged and publicly available (*21*), minimum connection times for a pair of flights can be found by drawing a vertical time slice at an airport runway and measuring the headway of the incoming and outgoing pair. In another example, the queue length of taxis waiting at a taxi stand at time t' can be found by counting the number of taxi trajectories that intersect a horizontal polygon time slice bounding the taxi stand at z-position $t = t'$.

CONCLUSION AND FUTURE WORK

This paper demonstrated how simple geometric techniques inspired by the time–space diagram can be used to efficiently compute and visualize performance metrics for a bus fleet. An app was developed that indexes observed bus trajectories as line segments in a time–space cube. Its intuitive interface allows users to analyze and visualize headways and spacing at any arbitrary point along a bus route.

The methods and algorithms described in this paper are general and can be applied to any transportation system that collects data with a time–space component. The methods and algorithms can be used as the basis for a visualization and analysis tool that will help transit operators and regulators to better understand network performance and behavior and to monitor service levels, network capacity, and delays.

The individual parts of this tool are well known in their respective fields: time–space diagrams in transportation, time–space cubes in human geography, and R-trees and intersection geometry in computer graphics. This work combined them in a novel way to analyze the real-world problem of bus bunching.

This tool can be used by transit operators and regulators to make better decisions when implementing control strategies or modifying bus frequency. Future versions will allow users to highlight common bunching spots and see how different control strategies or scenarios (e.g., increasing bus frequency) affect service levels. A future field trial conducted in collaboration with a local transit agency will illuminate agency needs and improve the app design and usability.

ACKNOWLEDGMENTS

This work was conducted at the Singapore MIT Alliance for Research and Technology (SMART) and the Singapore–ETH Centre for Global Environmental Sustainability. Support for this research was provided by the Future Urban Mobility project of the SMART Innovation Center Explorer. The authors thank Amedeo Odoni for his support and advice and the Singapore Land Transport Authority for making its data available.

REFERENCES

1. Daganzo, C. F. *Fundamentals of Transportation and Traffic Operations.* Emerald Group Publishing, Bingley, United Kingdom, 1997.
2. De Neufville, R., and A. R. Odoni. *Airport Systems: Planning, Design, and Management.* McGraw-Hill, New York, 2003.
3. Daganzo, C., and G. Newell. *Methods of Analysis for Transportation Operations.* Institute of Transportation Studies, University of California, Berkeley, 1995.

4. Koonce, P., L. Rodegerdts, K. Lee, S. Quayle, S. Beaird, C. Braud, J. Bonneson, P. Tarnoff, and T. Urbanik. *Traffic Signal Timing Manual.* FHWA-HOP-08-024. Kittelson and Associates, Portland, Ore., 2008.
5. Kraak, M.-J. The Space-Time Cube Revisited from a Geovisualization Perspective. *Proc., 21st International Cartographic Conference,* Durban, South Africa, 2003, pp. 1988–1996.
6. Ruan, M., and J. Lin. An Investigation of Bus Headway Regularity and Service Performance in Chicago Bus Transit System. *Proc., Transport Chicago Conference,* Chicago, Ill., 2009.
7. EZ-Link, Singapore. http://www.ezlink.com.sg/index.php.
8. Guttman, A. R-Trees: A Dynamic Index Structure for Spatial Searching. *ACM SIGMOD Record,* Vol. 14, No. 2, 1984, pp. 47–57.
9. Pfoser, D., C. S. Jensen, and Y. Theodoridis. Novel Approaches in Query Processing for Moving Object Trajectories. *Proc., 26th International Conference on Very Large Data Bases,* Cairo, Egypt, 2000, pp. 395–406.
10. Rasetic, S., J. Sander, J. Elding, and M. A Nascimento. A Trajectory Splitting Model for Efficient Spatio-Temporal Indexing. *Proc., 31st International Conference on Very Large Data Bases,* Trondheim, Norway, 2005, pp. 934–945.
11. Beckmann, N., H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. *ACM SIGMOD Record,* Vol. 19, No. 2, 1990, pp. 322–331.
12. Pei, J. *R-Tree Properties.* http://www.cs.sfu.ca/coursecentral/454/jpei/slides/r-tree.pdf.
13. Feng, W., and M. Figliozzi. Empirical Findings of Bus Bunching Distributions and Attributes Using Archived AVL/APC Bus Data. *Proc., 11th International Conference of Chinese Transportation Professionals,* Nanjing, China, 2011, pp. 4330–4341.
14. *Enhancing Public Transport.* Land Transport Authority, Singapore. http://bit.ly/1awzwgk.
15. Daganzo, C. *Why Bus Bunching Happens.* http://bit.ly/dkdxtp.
16. Newell, G. F., and R. B. Potts. Maintaining a Bus Schedule. *Proc., 2nd Australian Road Research Board Conference,* Melbourne, Australia, 1964, pp. 388–393.
17. Daganzo, C. F., and J. Pilachowski. Reducing Bunching with Bus-to-Bus Cooperation. *Transportation Research Part B: Methodological,* Vol. 45, No. 1, 2011, pp. 267–277.
18. Daganzo, C. F. A Headway-Based Approach to Eliminate Bus Bunching: Systematic Analysis and Comparisons. *Transportation Research Part B: Methodological,* Vol. 43, No. 10, 2009, pp. 913–921.
19. Xuan, Y., J. Argote, and C. F. Daganzo. Dynamic Bus Holding Strategies for Schedule Reliability: Optimal Linear Control and Performance Analysis. *Transportation Research Part B: Methodological,* Vol. 45, No. 10, 2011, pp. 1831–1845.
20. Zeng, W., C. Zhong, A. Anwar, S. M. Arisona, and I. V. McLoughlin. Metrobuzz: Interactive 3D Visualization of Spatiotemporal Data. *Proc., International Conference on Computer and Information Science,* Vol. 1, IEEE, New York, 2012, pp. 143–147.
21. *Flightstats: Global Flight Tracker and Status Tracking.* http://www.flightstats.com.

The Urban Transportation Data and Information Systems Committee peer-reviewed this paper.